В.М. ГЛУШКОВ

КИБЕРНЕТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА, ИНФОРМАТИКА

ИЗБРАННЫЕ ТРУДЫ

УДК 519.95

Кибернетика, вычислительная техника, информатика / Глушков В. М. Избранные труды: В 3 т. АН УССР. Ин-т кибернетики имени В. М. Глушкова; Редкол. В. С. Михалевич (отв. ред.) и др.— Киев : Наук. думка, 1990.— ISBN 5-12-001569-9.

Т. 2 : ЭВМ — техническая база кибернетики/ В. М. Глушков.— 1990.— 268 с. ISBN 5-12-001573-5.

Публикуются результаты исследований в области архитектурных принципов и облика перспективных ЭВМ, искусственного интеллекта, а также автоматизации проектирования, технологических аспектов создания и математического обеспечения современных ЭВМ.

Для специалистов в области кибернетики, вычислительной техники, прикладной математики, а также преподавателей, аспирантов и студентов вузов.

Ил. 7. Табл. 1.

Утверждено к печати ученым советом
Института кибернетики им. В. М. Глушкова
АН УССР

Редакция физики и кибернетики

Редактор *И. С. Кулаковская*

Г $\frac{1402070000\text{-}206}{M221(04)\text{-}90}$ 164-90

ISBN 5-12-001573-5 (т. 2)
ISBN 5-12-001569-9

Значение вычислительных машин и систем для научно-технического прогресса общества трудно переоценить. Электронные вычислительные машины, появившиеся в 50-х годах, т. е. существующие около 30 лет, являются очень динамично развивающимися и интенсивно усложняющимися техническими объектами. Об этом свидетельствует тот факт, что в настоящее время уже создаются ЭВМ V поколения. Отсюда следует, что время существования одного поколения ЭВМ — 6—7 лет. Кроме того, способность и возможности ЭВМ с точки зрения имитации в них мыслительной деятельности человека от поколения к поколению возрастают весьма существенно. Растет быстродействие и объем памяти, повышается уровень «машинного» интеллекта. В этих условиях вопросы теории построения ЭВМ, включающие в себя как представление об облике ЭВМ, так и методы и технологии разработки аппаратных и программных компонентов ЭВМ и ЭВМ в целом, а также технологические средства поддержки создания, проектирования и изготовления ЭВМ являются чрезвычайно актуальными.

В. М. Глушков много работал над указанными вопросами, и результаты его исследований, а также его научные идеи были по достоинству оценены у нас в стране и за рубежом, о чем свидетельствует общественное признание его работ.

Во второй том трехтомного издания включены работы В. М. Глушкова по формированию архитектурных принципов и облика ЭВМ. В частности, для ЭВМ новых поколений важное значение имеют идеи аппаратной поддержки языков высокого уровня, принципа макроконвейерной организации вычислений и многопроцессорной ЭВМ (раздел I), методы автоматизации совместного проектирования схемного и программного обеспечения ЭВМ, идеи реализации совместимости и преемственности поколений ЭВМ (раздел II), а также построение и использование ЭВМ как инструмента реализации идей искусственного интеллекта (раздел III).

На протяжении всего периода работы в области кибернетики и вычислительной техники В. М. Глушков занимался вопросами искусственного интеллекта, и многие его идеи в этом направлении еще не нашли своего воплощения.

Виктор Михайлович в исследовательской работе предпочитал воплощать принцип «единства далеких и ближних целей», который означает, что цель, к которой должен стремиться исследователь, состоит, по крайней мере, из двух составляющих: далекой, направленной в будущее (применительно к проблемам искусственного интеллекта — это создание «искусственного разума») и ближней, направленной

в настоящее (применительно к конкретным аспектам его деятельности здесь можно назвать реализацию эффективных алгоритмов аналитических преобразований, доказательство теорем в математике, распознавание зрительных образов и т. п.).

В. М. Глушков верил в достижение дальней цели в области искусственного интеллекта и в связи с этим даже высказывал очень интересные идеи о бессмертии человеческой личности. Результаты его исследований обычно завершались созданием новых средств вычислительной техники, которые по уровню своего «машинного» интеллекта превосходили предыдущие образцы.

Ю. В. Капитонова

РАЗДЕЛ I

АРХИТЕКТУРНЫЕ ПРИНЦИПЫ И ОБЛИК ЭВМ

ОБ ИНФОРМАЦИОННЫХ ВОЗМОЖНОСТЯХ СОВРЕМЕННЫХ ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

(Известия вузов. Электромеханика. — 1960.— № 7)

Современная наука и техника располагают большим арсеналом разнообразных средств для автоматизации вычислительной работы. Здесь и настольные клавишные автоматы, позволяющие автоматизировать выполнение арифметических операций над многозначными числами, и счетно-аналитические машины, автоматизирующие не только арифметические, но и некоторые логические операции (подборка и сортировка данных по тем или иным признакам), и, наконец, большая семья аналоговых (как механических, так и электронных) вычислительных машин, позволяющих решать сложные системы обыкновенных дифференциальных и алгебраических уравнений, а также многие типы уравнений в частных производных.

Однако наибольший интерес среди автоматических вычислительных устройств представляют электронные цифровые вычислительные машины с программным управлением и, в первую очередь, так называемые универсальные электронные цифровые машины, созданные в 1944—1946 гг. и ныне получившие достаточно широкое распространение.

Название «универсальные» эти машины получили не зря: по разнообразию типов решаемых на них задач современные большие электронные цифровые машины стоят вне конкуренции и намного превосходят все другие виды автоматических вычислительных устройств. С равным успехом их можно использовать при решении весьма широкого круга математических задач — от простейших бухгалтерских расчетов до решения сложных систем дифференциальных и интегральных уравнений.

Более того, современные универсальные электронные цифровые машины оказались пригодными для автоматизации многих процессов переработки данных, довольно далеких от математики. Достаточно указать здесь автоматический перевод с одного языка на другой, оркестровку музыкальных произведений, решение шахматных задач и т. п.

Такое огромное разнообразие применений приводит к постановке естественного вопроса: где же границы возможностей электронных вычислительных машин?

Можно ли хоть сколько-нибудь точно охарактеризовать то, что могут делать современные электронные вычислительные машины, и то, чего они не смогут сделать?

Для ответа на эти вопросы необходимо прежде всего хотя бы в общих чертах познакомиться с понятиями информации и ее преобразования.

Под информацией в современной науке принято понимать меру неоднородности распределения материи и энергии в пространстве и времени. При таком понимании информации оказывается возможным говорить, например, об информации, которую несет солнечный луч, шум горного обвала, шо-

ОБ ИНФОРМАЦИОННЫХ ВОЗМОЖНОСТЯХ СОВРЕМЕННЫХ ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

(Известия вузов. Электромеханика. — 1960.— № 7)

Современная наука и техника располагают большим арсеналом разнообразных средств для автоматизации вычислительной работы. Здесь и настольные клавишные автоматы, позволяющие автоматизировать выполнение арифметических операций над многозначными числами, и счетно-аналитические машины, автоматизирующие не только арифметические, но и некоторые логические операции (подборка и сортировка данных по тем или иным признакам), и, наконец, большая семья аналоговых (как механических, так и электронных) вычислительных машин, позволяющих решать сложные системы обыкновенных дифференциальных и алгебраических уравнений, а также многие типы уравнений в частных производных.

Однако наибольший интерес среди автоматических вычислительных устройств представляют электронные цифровые вычислительные машины с программным управлением и, в первую очередь, так называемые универсальные электронные цифровые машины, созданные в 1944—1946 гг. и ныне получившие достаточно широкое распространение.

Название «универсальные» эти машины получили не зря: по разнообразию типов решаемых на них задач современные большие электронные цифровые машины стоят вне конкуренции и намного превосходят все другие виды автоматических вычислительных устройств. С равным успехом их можно использовать при решении весьма широкого круга математических задач — от простейших бухгалтерских расчетов до решения сложных систем дифференциальных и интегральных уравнений.

Более того, современные универсальные электронные цифровые машины оказались пригодными для автоматизации многих процессов переработки данных, довольно далеких от математики. Достаточно указать здесь автоматический перевод с одного языка на другой, оркестровку музыкальных произведений, решение шахматных задач и т. п.

Такое огромное разнообразие применений приводит к постановке естественного вопроса: где же границы возможностей электронных вычислительных машин?

Можно ли хоть сколько-нибудь точно охарактеризовать то, что могут делать современные электронные вычислительные машины, и то, чего они не смогут сделать?

Для ответа на эти вопросы необходимо прежде всего хотя бы в общих чертах познакомиться с понятиями информации и ее преобразования.

Под информацией в современной науке принято понимать меру неоднородности распределения материи и энергии в пространстве и времени. При таком понимании информации оказывается возможным говорить, например, об информации, которую несет солнечный луч, шум горного обвала, шо-

ОБ ИНФОРМАЦИОННЫХ ВОЗМОЖНОСТЯХ СОВРЕМЕННЫХ ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

(Известия вузов. Электромеханика. — 1960.— № 7)

Современная наука и техника располагают большим арсеналом разнообразных средств для автоматизации вычислительной работы. Здесь и настольные клавишные автоматы, позволяющие автоматизировать выполнение арифметических операций над многозначными числами, и счетно-аналитические машины, автоматизирующие не только арифметические, но и некоторые логические операции (подборка и сортировка данных по тем или иным признакам), и, наконец, большая семья аналоговых (как механических, так и электронных) вычислительных машин, позволяющих решать сложные системы обыкновенных дифференциальных и алгебраических уравнений, а также многие типы уравнений в частных производных.

Однако наибольший интерес среди автоматических вычислительных устройств представляют электронные цифровые вычислительные машины с программным управлением и, в первую очередь, так называемые универсальные электронные цифровые машины, созданные в 1944—1946 гг. и ныне получившие достаточно широкое распространение.

Название «универсальные» эти машины получили не зря: по разнообразию типов решаемых на них задач современные большие электронные цифровые машины стоят вне конкуренции и намного превосходят все другие виды автоматических вычислительных устройств. С равным успехом их можно использовать при решении весьма широкого круга математических задач — от простейших бухгалтерских расчетов до решения сложных систем дифференциальных и интегральных уравнений.

Более того, современные универсальные электронные цифровые машины оказались пригодными для автоматизации многих процессов переработки данных, довольно далеких от математики. Достаточно указать здесь автоматический перевод с одного языка на другой, оркестровку музыкальных произведений, решение шахматных задач и т. п.

Такое огромное разнообразие применений приводит к постановке естественного вопроса: где же границы возможностей электронных вычислительных машин?

Можно ли хоть сколько-нибудь точно охарактеризовать то, что могут делать современные электронные вычислительные машины, и то, чего они не смогут сделать?

Для ответа на эти вопросы необходимо прежде всего хотя бы в общих чертах познакомиться с понятиями информации и ее преобразования.

Под информацией в современной науке принято понимать меру неоднородности распределения материи и энергии в пространстве и времени. При таком понимании информации оказывается возможным говорить, например, об информации, которую несет солнечный луч, шум горного обвала, шо-

рох листвы и т. п. Во всяком случае при этом необязательно требование осмысленности, с которым обычно неразрывно связывается понятие информации в ее обычном, житейском понимании.

При изучении способов представления информации наиболее целесообразным оказывается дискретный подход, заключающийся в том, что для информации любого данного вида вводится конечное число элементарных порций («квантов» или «атомов») информации, после чего любую порцию информации, воспринимаемую тем или иным объектом за конечный промежуток времени, можно представить в виде конечной последовательности элементарных порций.

В случае лексической информации в качестве элементарных порций («атомов») информации естественно рассматривать буквы (точнее, отдельные типографские знаки, включая знаки препинания, знак раздела и т. п.). По образцу этого частного примера и в других случаях элементарные порции информации называют буквами, а конечные последовательности таких букв — словами. Слова при этом могут иметь сколько угодно большую длину. Отметим, что в случае лексической информации при включении в основной алфавит знака раздела и знаков препинания словами можно считать не только обычные слова, но и целые абзацы, целые книги, как и отдельные части слов и, вообще, любые сочетания букв, не обязательно осмысленные.

В случае лексической информации выделение элементарных порций информации осуществляется относительно легко. Несколько сложнее это происходит при информации, циркулирующей в реальных физических процессах. Не вдаваясь в физические тонкости, связанные с квантовой механикой, отметим, что элементарные порции информации в физических процессах обычно определяются на основе изучения характеристик входных каналов устройств, воспринимающих эту информацию. При этом разрешающая способность входного канала определяет конечное число градаций воспринимаемой величины, которые и принимаются в качестве элементарных порций информации. Конечность же полосы пропускания канала дает возможность (на основании теоремы Котельникова) ввести дискретность во времени, так что за любой конечный промежуток времени устройство будет воспринимать конечную последовательность элементарных порций информации.

Резюмируя все сказанное, можно высказать своеобразный принцип «атомности» информации: **любую информацию, с которой приходится иметь дело в реальных преобразователях информации, можно представить в виде (конечного) слова в некотором конечном алфавите.**

Продемонстрируем этот принцип на каком-либо частном примере. Пусть, например, речь идет об информации, заданной в виде черно-белого полутонового рисунка. Для представления этой информации в виде слова в конечном алфавите проделаем следующее: прежде всего разобьем площадь рисунка системой горизонтальных и вертикальных линий на столь мелкие квадраты, чтобы их размеры были меньше, чем разрешающая способность устройства (например, человеческого глаза, воспринимающего эту информацию). Выразим среднюю яркость каждого из таких квадратов числом. Поскольку способности воспринимающего устройства к различению близких яркостей ограничены, то числа, измеряющие яркость квадратов, можно выбирать с ограниченным числом разрядов. Выписывая полученные значения яркостей в том или ином порядке (например, перебирая квадратики слова вправо и сверху вниз), получаем представление заданной информации в виде слова в конечном алфавите (алфавит

будет при этом состоять из цифр, занятой и знака раздела между различными числами).

Нетрудно видеть также, что с помощью довольно простых преобразований информацию, записанную в любом конечном алфавите $x_1, x_2, \ldots, x_n$, можно записать в алфавите, содержащем десять букв (цифр) или даже только две буквы (например, 0 и 1). Для этой цели достаточно обозначить все буквы исходного алфавита попарно различными комбинациями символов 0 и 1. Таким образом, способ представления информации в виде чисел и даже в виде так называемых двоичных чисел является универсальным, а именно: с помощью весьма простых преобразований любую информацию можно представить (закодировать) числами и обратно, зная закон кодирования, по числовой форме представления информации можно восстановить ее исходную форму.

Более того, современная техника уже располагает приборами, позволяющими автоматически, без участия человека, преобразовывать практически любой вид информации в цифровую форму. Действительно, техника сегодняшнего дня способна (по крайней мере в принципе) преобразовать информацию любого вида в информацию в виде изменяющегося со временем электрического напряжения (телефония, телевидение и др.). Вместе с тем уже разработаны специальные приборы (так называемые цифровые вольтметры), которые автоматически измеряют электрическое напряжение и превращают его в цифровой код. Правда, частные характеристики современных цифровых вольтметров таковы, что не допускают непосредственного их сопряжения, например, с телевизионной аппаратурой без существенного снижения темпа телевизионной развертки. Однако в данном случае речь идет уже о чисто технических, а не о принципиальных ограничениях.

Аналогично обстоит дело с обратными преобразованиями информации (из цифровой формы в произвольную). Таким образом, резюмируя сказанное, приходим к выводу, что числовая форма представления информации, с которой оперируют электронные вычислительные машины, является универсальной и (по крайней мере в принципиальном отношении) не ограничивает возможностей этих машин как универсальных преобразователей информации.

Общую схему преобразования информации с помощью электронной вычислительной машины можно описать следующим образом. Пусть с помощью преобразования φ исходная информация A_1 преобразуется в некоторую информацию A_2. Обозначим через α_1 процесс кодирования (выражения в цифровой форме) для информации A_1, а через α_2 — для информации A_2. Результат кодирования для информации A_1 обозначим через B_1, а для информации A_2 — через B_2. Кодирующие преобразования α_1 и α_2 предполагаются обратными, иначе говоря из закодированной информации B_1 и B_2 с помощью преобразований $\beta_1 = \alpha_1^{-1}$ и $\beta_2 = \alpha_2^{-1}$ можно восстановить информацию A_1 и A_2 соответственно. Легко видеть, что преобразование φ при заданных α_1 и α_2 индуцирует преобразование $\psi = \alpha_1^{-1}\varphi\alpha_2$ в информацию B_2. Обратно, задавая преобразование ψ по формуле $\varphi = \alpha_1 \psi \alpha_2^{-1}$, получаем преобразование φ. Таким образом, правила преобразования информации произвольного вида индуцируют некоторые правила преобразования числовой информации и наоборот.

Помня об огромном качественном разнообразии правил переработки информации, трудно, на первый взгляд, надеяться реализовать все эти правила в одной машине, обеспечив тем самым универсальность этой ма-

шины. Действительно, что общего между правилами решения математических задач и правилами, по которым сочиняются музыкальные мелодии, между правилами игры в шахматы и правилами перевода с одного языка на другой? Однако это общее существует. Оказывается, любое преобразование числовой информации в числовую, выполняемое на основе произвольной системы точно сформулированных правил (алгоритма), можно составить из небольшого числа заранее фиксированных элементарных правил, комбинируемых различным образом в (конечные) последовательности. При этом каждое из элементарных правил будет встречаться в последовательности, вообще говоря, много раз, так что из небольшого числа типов исходных элементарных правил можно составлять сколь угодно длинные последовательности.

Систему элементарных правил преобразований информации, обеспечивающих возможность построения любого алгоритма, естественно называть алгоритмически полной. Оказывается, среди всех автоматов-преобразователей информации, созданных до сих пор человеком, лишь универсальные электронные цифровые машины обладают свойством алгоритмической универсальности. Более подробный анализ показывает, что алгоритмическая универсальность современных электронных цифровых машин обеспечивается возможностью последовательного многократного выполнения (в любом заданном порядке) всего лишь трех операций, а именно операций переадресации, условного перехода и произвольных пересылок информации из одной ячейки памяти в другую.

Электронная цифровая машина, в набор операций которой входят три указанные операции, а также операции, обеспечивающие ввод и вывод информации, в принципе оказывается способной выполнять любые преобразования информации, не взирая на то, будут ли правила, определяющие это преобразование, иметь математическую или какую угодно иную природу. Чтобы заставить машину выполнять то или иное преобразование информации (решение математической задачи, перевод с одного языка на другой и т. п.), необходимо лишь представить правила соответствующего преобразования в виде последовательности элементарных правил, каждое из которых может быть выполнено машиной, и ввести полученную последовательность (так называемую программу) в память машины.

В этом вопросе обнаруживается слабое место современных электронных вычислительных машин. Дело в том, что быстродействующая (так называемая оперативная) память современных электронных цифровых машин относительно невелика по объему и, следовательно, не может вместить в себя слишком длинные программы. Данное положение частично компенсируется тем, что вычислительные машины кроме оперативной памяти обычно снабжаются также более медленной (так называемой внешней) памятью на магнитных барабанах, магнитных лентах и т. п. Внешняя память имеет практически неограниченный объем, но зато при работе с ней вычислительная машина резко снижает скорость переработки информации.

Таким образом, границы возможностей современных электронных вычислительных машин определяются прежде всего их количественными характеристиками (объемом памяти, скоростью работы, пропускной способностью вводных и выводных каналов). Поэтому для оценки возможностей универсальной вычислительной машины целесообразно сравнить ее количественные характеристики с соответствующими характеристиками естественного универсального преобразователя информации, каким является мозг человека.

Для проведения сравнительных оценок удобно пользоваться количественной оценкой информации в специальных единицах — битах. Информацию в один бит несет один двоичный разряд (различие между двумя равновероятными возможностями). Поскольку с помощью n двоичных разрядов можно составить 2^n различных комбинаций нулей и единиц, то выбор между 2^n равновероятными возможностями дает информацию в n битов. Так, в тексте, написанном с помощью $32 = 2^5$ различных букв, каждая буква несет информацию в 5 битов.

Сравнительный анализ возможностей современных электронных вычислительных машин и мозга человека показывает прежде всего огромное различие в объемах памяти. В то время как в лучших современных электронных цифровых машинах объем оперативной памяти не превышает $3 \cdot 10^6$ бит, то, по самым скромным оценкам, память человека оценивается числом 10^{12}—10^{13} бит, т. е. на шесть порядков больше. Это приводит к тому, что человек способен накопить в своей памяти большой объем информации, позволяющий ему реализовать весьма сложные и разнообразные правила переработки информации.

Современные электронные вычислительные машины значительно уступают человеку в пропускной способности каналов, осуществляющих ввод информации. В то время как лучшие системы ввода с перфокарт перфолент обеспечивают ввод со скоростью, не превосходящей $5 \cdot 10^3$ бит/с, пропускная способность всех органов чувств человека оценивается числом порядка $2 \cdot 10^7$ бит/с, причем почти половина этого числа приходится на орган зрения. Использование магнитных лент и быстродействующих читающих устройств повышает скорость ввода в машину примерно на порядок, но даже в этом случае пропускная способность вводных каналов для машины оказывается на 2—3 порядка меньше, чем у человека. Однако отметим, что в ряде случаев, например при чтении печатного текста, пропускная способность органа зрения человека используется крайне неэффективно; при скорости чтения в одну страницу за минуту скорость ввода информации не превышает 200 бит/с, что значительно меньше соответствующих показателей машины.

Аналогичное свойство обнаруживается и в выводных каналах: хотя пропускная способность нервных путей, управляющих движениями мышц человека, довольно высока, однако скорость вывода лексической информации даже в случае стенографической скорости письма намного уступает скорости вывода информации быстродействующими буквопечатающими устройствами современных вычислительных машин (эта скорость в лучших устройствах достигает 10^5 бит/с).

Наиболее заметно преимущество машины перед человеком в скорости и точности переработки информации. Так, при выполнении арифметических операций над многозначными числами лучшие современные электронные цифровые машины превосходят по скорости возможности человека в несколько миллионов раз. В то же время следует подчеркнуть, что при выполнении некоторых других операций преимущество машины заметно уменьшается, а в ряде случаев (например, при распознавании зрительных образов) исчезает совсем. Тем не менее при решении большого числа важных задач, связанных с переработкой значительного объема информации, преимущество в скорости, обеспечиваемое машиной, является весьма заметным. На этом, собственно, и основаны многочисленные применения электронных вычислительных машин, которые оказываются незаменимыми всякий раз, когда пропускная способность человека как преобразователя информации становится недостаточной. В качестве при-

мера можно привести некоторые научные и инженерные задачи, для решения которых человеку, не вооруженному машиной, потребовались бы десятки тысяч лет и которые решаются с помощью машины за сутки. Такими являются, например, некоторые задачи атомной физики, задачи оптимального планирования перевозок и многие другие.

Электронные вычислительные машины оказываются совершенно необходимыми при построении сложных систем учета и планирования, при управлении быстродействующими процессами, процессами, зависящими от большого числа разнообразных факторов, и во многих других областях.

В заключение отметим, что все приведенные выше показатели относятся к машинам сегодняшнего дня. Вряд ли можно сомневаться, что в ближайшем будущем электронные цифровые машины намного превысят эти показатели. Не исключено также, что наряду с вычислительными машинами, построенными на основе принципа программного управления, появятся другие типы универсальных преобразователей информации, логическая структура которых будет более близкой к структуре мозга человека и которые сохранят вместе с тем основные достоинства (скорость, точность работы) современных электронных вычислительных машин.

ПРОБЛЕМЫ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ

(Сб. тр. конф. «Вопросы вычислительной техники», Киев.— 1960)

Появление электронных цифровых машин с программным управлением и успешное развитие теории и практики программирования представляют собой крупное достижение в области вычислительной техники и математики. Впервые в истории автоматики цифровые машины с программным управлением достигли операционной универсальности. Это означает, что любые точные правила (не обязательно математической природы) преобразования числовой информации могут быть разложены на элементарные операции, входящие в систему операций любой современной универсальной цифровой машины.

Поскольку в числовой форме относительно легко представляется информация любого вида, то с принципиальной точки зрения возможности уже существующих вычислительных машин, как преобразователей информации, ничем не ограничены. Вместе с тем хорошо известно, что в действительности существуют различные ограничения, связанные с надежностью машины, объемом ее памяти, скоростью работы, не говоря уже об алгоритмической неизученности многих видов преобразования информации. Всякий новый этап в преодолении указанных ограничений приводит к необходимости решать многие трудные научные и технические проблемы. Разумеется, в одной статье невозможно охарактеризовать сколько-нибудь полно все проблемы, стоящие в настоящее время перед вычислительной техникой и вычислительной математикой. Поэтому мы ограничимся рассмотрением лишь некоторых важнейших проблем, связанных с дальнейшим развитием и использованием электронных цифровых автоматов.

Одной из первых должна быть названа проблема создания теории синтеза схем электронных цифровых машин. Эта проблема имеет три основных аспекта: алгоритмический, комбинационный и надежностный. Алгоритмический синтез цифрового автомата включает в себя вопросы, связанные с функционированием автомата времени. Этапу алгоритмического синтеза предшествует этап выбора класса задач, который должен решаться синтезируемым автоматом, включая выбор способа кодирования входной и выходной информации.

В наиболее общем виде проблема алгоритмического синтеза цифрового автомата сводится к проблеме представления полугруппы подстановками. Действительно, пусть $a_1, a_2, \ldots, a_k$ — различные состояния входных цепей автомата. Обозначим последовательными натуральными числами $1, 2, \ldots, m$ все возможные состояния выходных цепей автомата. Тогда исходными данными для синтеза автомата будет конечная или бесконечная система равенств вида $a_{i_1}, a_{i_2}, \ldots, a_{i_s} = r$, где r — одно из чисел $1, 2, \ldots, m$. Каждое из таких равенств означает, что, будучи приведенным в то или иное состояние (например, состояние, соответствующее выходу 1), автомат должен последовательностью $a_{i_1}, a_{i_2}, \ldots, a_{i_s}$ состояний его входов пере-

водиться в состояние, соответствующее выходу r. Добавляя в случае необходимости к состояниям 1, 2, ..., m дополнительные состояния автомата $m+1$, ..., $m+n$, всегда можно для каждого состояния входа a_i найти такую подстановку A_i (отображение в себе) системы чисел 1, 2, ..., $m+n$, что для каждого определяющего соотношения $a_{i_1}, a_{i_2}, \ldots, a_{i_s} = r$ подстановка $A_{i_1}A_{i_2}A_{i_s}$ переводит число 1 в число r. После того как найдены подстановки для всех состояний входов $a_1, a_2, \ldots, a_k$, дальнейший синтез автомата носит уже чисто комбинационный характер, т. е. сводится к схемной реализации тех или иных булевых функций, уже не зависящих от времени.

Основной проблемой, возникающей при таком общем подходе к синтезу автомата, является проблема его схемной минимизации. В полной мере эта проблема не может быть решена на этапе алгоритмического синтеза, однако для ряда случаев представляет интерес минимизация числа дополнительных состояний автомата $m+1, \ldots, m+n$. В такой ограниченной постановке проблема минимизации может и должна решаться на первом этапе синтеза.

Несмотря на важность решения проблемы алгоритмического синтеза в общем виде, не менее, а может быть и более важными являются планомерные поиски частных алгоритмических схем организации цифровых автоматов. Речь идет здесь прежде всего об автоматах с программным управлением, алгоритм работы которых определяется не только схемой, но и той или иной информацией об алгоритме (программой), вводимой в автомат извне. Как известно, именно такие частные алгоритмические схемы определили прогресс вычислительной техники за последние 10 лет.

В настоящее время найдены и успешно используются две основные алгоритмические схемы организации цифровых автоматов: цифровые дифференциальные анализаторы и операционно-адресные машины, оперирующие словами фиксированной длины. Несмотря на то что в операционно-адресных машинах обычно предусматривается ряд логических операций, они тем не менее оказываются плохо приспособленными к выполнению сложных неарифметических алгоритмов. В связи с этим весьма актуальной является задача нахождения новых алгоритмических схем автоматов прежде всего для преобразования информации языкового типа (например, для целей перевода с одного языка на другой). Важное значение имеют также различные алгоритмические схемы для преобразования зрительной информации в словесную.

Есть основания надеяться, что алгоритмические схемы, получаемые при решении указанных частных задач, могут оказаться хорошо приспособленными для реализации довольно широкого круга неарифметических алгоритмов. Чрезвычайно перспективной является проблема создания алгоритмических схем автоматов, основанных на аналогиях с нервной системой высших животных и человека. Такие схемы могут строиться с использованием случайных связей и последующей их организацией в результате специального процесса обучения.

Большой интерес представляют алгоритмические схемы, основанные на объединении принципов цифровых дифференциальных анализаторов и универсальных (операционно-адресных) машин с программным управлением.

Необходимо разработать методику, позволяющую осуществлять формальные эквивалентные преобразования различных схем управления, применяющихся в современных цифровых машинах. Прежде всего это относится к схеме микропрограммного управления, где представляется

возможным построить полную систему элементарных эквивалентных преобразований.

Последняя задача является переходом к проблеме комбинационного синтеза. Несмотря на большое число работ, посвященных проблеме комбинационного синтеза, она не может пока считаться сколько-нибудь удовлетворительно решенной даже для простейшего случая контактных схем. Существующие способы минимизации контактных схем имеют, за малым исключением, рецептурный характер и содержат слишком большой элемент перебора. В связи с этим значительный интерес приобретает использование вычислительных машин для преобразования и упрощения булевых матриц.

При переходе контактных схем к более сложным электронным схемам аппарат булевых функций оказывается, как правило, уже недостаточным. Успех применения булевых функций к контактным схемам обеспечивается тем, что управляемый релейный контакт представляет собой двоичный элемент, близкий к идеальному: даже несколько десятков тысяч включенных параллельно разомкнутых контактов будут иметь гораздо большее сопротивление, чем такое же количество включенных последовательно замкнутых контактов. При использовании электронных схем подобное явление уже не наблюдается, приходится считаться с искажением сигналов даже в относительно коротких цепях, с временными задержками, нагрузочными характеристиками элементов и т. п.

Разумеется, трактуя сигналы как функции, а элементы схемы — как операторы, преобразующие эти функции (и, в свою очередь, зависящие от схемы), можно построить теорию синтеза, достаточно точно отображающую действительную ситуацию. Однако нелинейность большинства цепей электронных цифровых машин делает такой метод чрезвычайно громоздким и практически непригодным. В то же время хорошо известно, что сигналы в цифровых машинах могут варьироваться в достаточно широких пределах, не приводя к каким бы то ни было погрешностям в работе машин. Естественно, возникает мысль задавать критерий качества сигнала с помощью конечного числа условных символов-оценок, например: хороший нуль, плохой нуль, хорошая единица, плохая единица. Используя такое задание сигнала и заведомо огрубляя (в сторону ухудшения) разработку схемы, отдельные ее элементы можно представлять вещественными функциями или даже функциями многозначной логики. При этом основная проблема состоит в определении полных систем элементарных преобразований систем функций, описывающих ту или иную схему, и алгоритмов минимизации схем, оперирующих такими преобразованиями.

Говоря кратко, проблема состоит в том, чтобы построить описания электронных схем (работающих по двоичному принципу), промежуточные между описанием с помощью булевых функций и описанием, задающимся дифференциальными уравнениями электрических цепей схемы. Такое промежуточное описание должно быть достаточно простым и, вместе с тем, учитывать необходимость введения в схему тех или иных восстанавливающих элементов (катодных повторителей, усилителей, формирователей и т. п.), не имеющих никакого смысла на уровне описания, задаваемого булевыми функциями.

Чрезвычайно важна проблема построения теории надежности электронных цифровых машин. Эта проблема имеет два аспекта: статический и динамический. В статическом аспекте теория надежности электронных цифровых машин не представляет собой чего-либо принципиально нового по сравнению с теорией надежности электронной аппаратуры вообще.

Здесь речь идет о выработке методики расчета параметров схемы, обеспечивающих возможно большой учет изменяющихся с течением времени номиналов без заметного ухудшения качества работы. К этому же аспекту относятся задачи определения среднего времени безаварийной работы машины, нахождения методики испытания и профилактики и т. п. В указанном направлении уже много сделано, однако законченной теории применительно к электронным цифровым машинам пока еще не создано.

Еще менее разработаны проблемы динамической надежности электронных цифровых машин. Эти проблемы в подавляющем большинстве являются принципиально новыми и не имеют соответствующих аналогов в области обычной электронной аппаратуры. Речь идет, прежде всего, о проблеме сбоев под влиянием шумов.

Как известно, работа всякого электронного элемента сопровождается большими или меньшими шумовыми эффектами. Даже в случае, когда средняя амплитуда шума весьма мала, существует (хотя и весьма малая) вероятность больших выбросов шумов.

Обычная электронная аппаратура обладает большой избыточностью информации, позволяющей игнорировать влияние подобного рода факторов. Если, например, шумовые выбросы подавляют или сильно искажают в среднем каждый миллиардный импульс генератора развертки в телевизоре, то высказываемое таким шумом искажение изображения может даже не быть сколько-нибудь заметным. Телевизор будет при этом считаться вполне исправным. В то же время исчезновение в среднем одного из каждого миллиарда управляющих импульсов в современной скоростной электронной цифровой машине приведет к столь частым сбоям, что сделает практически невозможным решение на ней сколько-нибудь сложных задач.

В связи со сказанным актуальными являются задачи изучения процессов возникновения и усиления искажений в длинных цепях циркуляции и рециркуляции импульсов и нахождение методов определения вероятности сбоев в электронных цифровых автоматах. Простейшей из таких задач является определение среднего времени циркуляции импульса в динамическом триггере (усилителе с линией задержки) при заданном уровне шумов и неизменности всех параметров триггера и питающего напряжения.

Объединение всех трех аспектов теории синтеза электронных цифровых автоматов (алгоритмического, комбинационного и надежностного) позволяет решать задачу синтеза и минимизации схем на основании единого общего критерия.

В качестве такого критерия наиболее естественно выбрать критерий цены эффективного быстродействия, который определяется следующим образом: с каждым элементом машины сопоставляется некоторое число — «цена» элемента. При этом в понятие цены включается не только обычная продажная цена элемента, но и его габариты: вес, потребление энергии, срок службы, частота и сложность ремонта, а также степень унификации элементов. Из цены отдельных элементов можно составить представление о цене z эксплуатации машины в единицу времени (включая амортизационные расходы).

Пусть машина решает класс задач с известной заранее статистикой элементарных операций. Тогда можно говорить о быстродействии машины, понимая под этим среднее (при данной статистике) число N элементарных операций, выполняемых машиной в единицу времени. Величину $\frac{z}{N}$ естественно назвать ценой быстродействия, однако она еще не в полной мере характеризует качество машины. Действительно, недостаточная надеж-

ность машины (задаваемая вероятностью сбоя) приводит к необходимости проведения многократных расчетов для получения правильного результата с одной и той же фиксированной заранее вероятностью $1 - \varepsilon$. Таким образом, эффективное быстродействие машины будет составлять лишь некоторую часть $k < 1$ теоретического быстродействия, а цена эффективного быстродействия будет выражаться дробью $\frac{z}{kN}$.

Умея синтезировать и минимизировать комбинационные схемы по заданной алгоритмической схеме, а также находить вероятность сбоя автомата и его эффективное быстродействие, можно ставить задачу минимизации критерия цены эффективного быстродействия за счет выбора подходящей алгоритмической схемы автомата. Интересно также решить задачу использования критерия цены эффективного быстродействия отдельно на этапах алгоритмического и комбинированного синтеза.

Из технических проблем в развитии современной вычислительной техники наиболее актуальной по-прежнему остается проблема создания быстродействующих запоминающих устройств большой емкости. При решении этой задачи возникают две основные проблемы переключения и технологичности. Действительно, если мы умеем изготовлять одну ячейку памяти, то, по крайней мере с принципиальной стороны, ничто не может помешать изготовлению многих миллионов таких ячеек. Однако в случае необходимости индивидуального изготовления и подключения трудности технологического порядка ограничивают значительное увеличение объема запоминающих устройств.

При решении проблемы переключения ячеек запоминающих устройств в настоящее время используют два основных способа, которые можно назвать матричным и логарифмическим. Первый способ применяется в системах памяти на магнитных сердечниках: сердечники располагаются по площади прямоугольника (матрицы), а выборка производится с помощью переключения двух систем проводов, параллельных сторонам прямоугольника и пронизывающих сердечники. При квадратном расположении n сердечников для доступа к любому из них достаточно выполнить $2\sqrt{n}$ переключений.

Логарифмический принцип переключения в чистом виде можно применить в электронно-лучевых трубках. Предположим, что электронный луч проходит последовательно через m пар отклоняющих пластин, на каждую из которых подается одно из двух фиксированных напряжений. Если система рассчитана так, что переключение напряжения на первой паре пластин вызывает отклонение луча на $^1/_2$ экрана, переключение напряжения на второй паре отклоняет луч на $^1/_4$ экрана и т. д., то с помощью переключения m выводов от пластин можно получить $2m$ позиций переключения луча. Число необходимых переключений в этом случае равно двоичному логарифму от числа ячеек памяти.

Интересной является задача объединения двух описанных способов переключения, для чего необходимо выходы матрицы коммутировать с помощью электронного луча. Большие токи, требуемые для переключения ячеек в ферритовых матрицах, практически исключают возможность помещения таких матриц внутри электронно-лучевой трубки. Однако для матриц, построенных на сегнетоэлектриках, такая возможность не исключена. Если бы удалось получать тонкие пленки сегнетоэлектриков, однородные на больших площадях, то можно было бы относительно легко собрать куб из пластин диэлектрика с нанесенной на них системой параллельных металлических полосок, покрытой сверху слоем сегнетоэлектрика.

Поворачивая под прямым углом системы полосок на соседних пластинах и выводя концы полосок на торцы куба, нетрудно было бы получить систему памяти, содержащую многие миллионы ячеек. Помещая эту систему внутрь электронно-лучевой трубки, можно было бы осуществить произвольное коммутирование ячеек с помощью переключения всего лишь нескольких десятков выводов от отклоняющих пластин.

Самостоятельный интерес представляет также проблема построения систем пассивной быстродействующей памяти большого объема. По-прежнему актуальной остается задача увеличения быстродействия электронных цифровых машин. Для решения этой задачи необходимо не только увеличивать быстродействие элементов, но и создавать новые алгоритмические схемы переработки информации, обеспечивающие более короткие цепи логических передач, параллельное выполнение операций и т. п.

Из проблем в области новых типов вводных и выводных устройств особый интерес представляет проблема создания универсальных читающих автоматов. Существующие в настоящее время читающие устройства нацелены на решение важной, но тем не менее достаточно узкой утилитарной задачи по считыванию печатного буквенного и числового текста. Между тем нетрудно представить себе систему, состоящую из универсальной вычислительной машины и передающей телевизионной трубки, обладающую гораздо большей универсальностью. При этом телевизионная трубка должна быть снабжена дискретной системой развертки, позволяющей установить считывающий луч по любому заданному адресу, и устройством для преобразования в цифровой код напряжения, индуцированного яркостью считываемой точки экрана. В таком случае машина может по специальной программе управлять движением луча и производить расшифровку не только букв и цифр, но также простейших (прежде всего контурных) рисунков, чертежей и т. п.

Большой интерес представляет создание управляемого импульсными кодами универсального выводного устройства, работающего по принципу «руки» атомного манипулятора, и управляемого универсальной вычислительной машиной. Необходима также разработка надежных специализированных устройств для ввода и вывода из вычислительной машины графиков, заданных тем или иным стандартным образом (например, на кинопленте).

Большое значение для убыстрения сроков проектирования новых вычислительных машин имело бы решение проблемы автоматизации и нормализации основных узлов машин (электронных и электромеханических ЗУ, арифметических устройств и т. д.), а также стандартизация связей между узлами. Аналогичная задача может быть поставлена для вводных и выводных устройств, в том числе для непрерывно-дискретных и дискретно-непрерывных преобразователей.

Решение задачи нормализации узлов (включая преобразователи) позволило бы относительно просто решить задачу создания нормального ряда машин в целях изучения производственных процессов и управления ими.

В области использования уже существующих электронных цифровых машин также имеется много важных и интересных проблем. Одной из первых следует назвать проблему создания универсального алгоритмического языка, пригодного для простого удобного описания сложных неарифметических алгоритмов типа программирующих программ, программ перевода, игровых алгоритмов и т. п. Такой язык должен, с одной стороны, не зависеть от конкретных особенностей той или иной машины,

а с другой — допускать простое преобразование в язык конкретных машин. Разработка универсального алгоритмического языка должна сопровождаться изучением эквивалентных преобразований алгоритмов, записанных с его помощью. Без решения этих задач трудно рассчитывать, что программирование явится предметом содержательной математической теории.

Второй важнейшей задачей является задача, которую можно образно назвать «обучение машины математике». Речь идет о том, чтобы упорядочить алгоритмы вычислительной математики, включая алгоритмы буквенных преобразований и алгоритмы автоматического программирования, и разработать такую систему кодирования информации, которая обеспечила бы непосредственное использование выходной информации одних алгоритмов в качестве входной информации последующих алгоритмов.

В случае наличия достаточно емких запоминающих устройств оказалось бы возможным хранить все множество упорядоченных алгоритмов (записанных с помощью универсального языка или в какой-нибудь иной экономной форме) в машине. На первом этапе это дало бы возможность осуществлять управление машиной с помощью «крупноблочных» приказов («преобразовать дифференциальное уравнение в уравнение в конечных разностях», «решить систему алгебраических уравнений» и т. п.) на более высоком уровне, чем это возможно при использовании библиотеки стандартных подпрограмм. Последовательное же развитие указанного направления в сочетании с разработкой читающих устройств может привести к полной автоматизации процесса решения задач вычислительной математики на электронных цифровых машинах и полностью упразднить труд программистов.

Несколько обобщая постановку вопроса, придем к проблеме ступенчатой организации системы алгоритмов. Помимо алгоритмов первой системы, осуществляющих непосредственное преобразование входной информации и приспособленных для последовательной работы один за другим, вводятся алгоритмы высших ступеней, управляющие порядком выполнения алгоритмов более низких ступеней. В этом же круге проблем лежит построение систем самосовершенствующихся алгоритмов: кроме алгоритмов, управляющих порядком выполнения алгоритмов низших ступеней, здесь появляются также алгоритмы, изменяющие те или иные алгоритмы низших ступеней.

Отметим, что обычно применяемое в математике понятие алгоритма нуждается в известном расширении за счет отказа от свойства результативности и допущения случайных переходов. При этом создаются гораздо большие возможности для практической алгоритмизации процессов творческого характера, прежде всего для процессов поиска доказательств новых математических теорем.

Решению этой последней проблемы должно предшествовать решение проблемы формализации смысловых связей в языке (в той или иной узкой области) на основании детального изучения доказательств математических теорем в связи с их языковой структурой (лингвоматематика).

Мы привели только некоторые из важнейших задач, оставив в стороне такие большие самостоятельные проблемы, как численный анализ, проблемы аналоговых машин и их сочетаний с дискретными, проблемы применения вычислительных машин для управления производственными процессами и многие другие. Однако сказанного достаточно, чтобы понять, какие важные задачи стоят перед современной вычислительной техникой и вычислительной математикой и какие грандиозные возможности открываются для их дальнейшего прогресса.

ПРИНЦИПЫ ПРОЕКТИРОВАНИЯ СОВРЕМЕННЫХ СИСТЕМ ОБРАБОТКИ ДАННЫХ И ИНФОРМАЦИОННО-СПРАВОЧНЫХ СИСТЕМ

(Информационные материалы.— № 8 (21). — 1968)

Как известно, в 1962—1963 гг. произошло серьезное изменение принципов, на которых основывается использование электронных вычислительных машин. Если до этого времени основную массу машинного времени установленных универсальных ЭВМ занимало решение эпизодических, часто не связанных друг с другом, задач, то теперь основное внимание при решении большого числа классов сложных задач уделяется системному подходу. Особенностью этого подхода является прежде всего наличие комплекса взаимосвязанных программ, пользующихся общими массивами информации и не требующих многократных вводов и выводов промежуточных данных. Выходные данные в системах оформляются в виде окончательных документов, не требующих никакой последующей обработки. Сбор входных данных в значительный степени автоматизируется, принимаются специальные меры против их ненужного дублирования.

При этом в каждой подобной системе обработки данных достаточно четко выделяются две части: справочно-информационная и собственно вычислительная или обрабатывающая. В задачу программ справочно-информационной части входит подготовка данных (выбираемых из общих информационных массивов) для решения тех или иных задач обработки данных, а также пополнение и обновление общих информационных массивов. Подобные справочно-информационные системы оказываются, вообще говоря, существенно более сложными, чем обычные справочно-информационные системы предыдущего периода, основной задачей которых был поиск необходимых данных в большом массиве информации по тому или иному их описанию. Задача усложняется еще и тем, что справочно-информационные подсистемы современных систем обработки данных работают обычно в режимах, обеспечивающих максимальное использование установленного оборудования (мультипрограммирование) и возможность параллельного обслуживания большого числа абонентов в режимах разделения времени.

Да и «чистые» справочно-информационные системы начинают постепенно перерастать в сложные многопрограммные системы обработки данных. Рассмотрим, например, классические научные справочно-информационные системы, обслуживающие ту или иную отрасль знаний. Не успев еще родиться, такие системы нуждаются в существенном расширении и усовершенствовании. Особенно серьезно эта проблема стоит в дедуктивных науках, в первую очередь — в математике. Сегодня уже ясно чувствуется необходимость в своеобразном симбиозе справочно-информационных систем в математике с системами автоматизации доказательств теорем.

Такая система наряду с рефератами статей по тому или иному разделу математики должна содержать некоторую программу логического

вывода — машинный «алгоритм очевидности» — и специальную операционную систему программирования доказательства. Система должна иметь много пультов для работы в режиме разделения времени. Реферат каждой новой статьи вводится с одного из таких пультов в специальном входном языке. Должны быть введены определения всех новых понятий, встречающихся в статье, формулировки и доказательства теорем. Степень подробности доказательства зависит от силы машинного алгоритма очевидности: необходимо, чтобы машина, пользуясь этим алгоритмом, могла восстановить опускаемые части доказательства.

В режиме запросов машина должна не только устанавливать новизну того или иного результата, но и определять, не является ли он очевидным следствием (с точки зрения машинного алгоритма очевидности) уже известных машине результатов. При программировании доказательств возможности алгоритма очевидности расширяются указаниями типа «использовать конструкции из доказательства таких-то известных теорем» или прямым описанием класса конструкций и промежуточных целей.

Еще одним примером сложной справочно-информационной системы является система, предложенная автором в 1964 г. в процессе разработки проекта Единой государственной системы вычислительных центров. Эта система должна охватывать экономику всей страны и иметь многоярусную иерархическую структуру. В нижнем ярусе располагаются около пяти тысяч первичных вычислительных центров, обслуживающих крупные предприятия и учреждения либо группы мелких и средних предприятий и учреждений.

Задача этих центров — двойная. С одной стороны, они накапливают постоянно обновляемые массивы данных, характеризующих обслуживаемые ими предприятия: основные фонды, запасы материалов, кадры, финансы, планы и т. п. Данные эти обновляются за счет машинных дубликатов первичных документов (накладных, паспортов нового оборудования, записей в книгу учета кадров и т. д.), которые должны изготовляться без дополнительной затраты труда в момент составления первичных документов. Для этой цели используются флексорайтеры, телетайпы, прямо соединенные с ЭВМ, записи магнитными чернилами, карты с карандашными отметками, специальные кассовые аппараты и другое оборудование.

С другой стороны, задача первичных центров состоит в решении тех или иных задач для обслуживаемых ими предприятий, включая задачу получения справочной информации от других первичных центров и ее обработки. При этом предполагается, что вся информация в первичных центрах содержится в виде, удобном для быстрого поиска и передачи в линии связи. Основным носителем информации при этом могли бы быть магнитные ленты, для особо часто требуемой информации — магнитные диски и барабаны.

Первичные центры создаются не только в промышленности, но и на транспорте, в строительстве, в сельском хозяйстве, в торговле, в различных государственных учреждениях (в частности, во всех министерствах).

Второй ярус системы составляют мощные территориальные вычислительные центры, дислоцированные в крупных промышленных центрах, являющихся узловыми точками общегосударственной системы связи. Общее число их — 50—60. Каждый такой центр обслуживает определенную территорию и соединен с любым первичным центром на этой территории телефонным каналом связи.

Друг с другом территориальные центры соединяются широкополосными каналами связи, позволяющими осуществлять обмен информаци-

ей со скоростью, принятой для магнитных лент ЭВМ (60—120 кГц). Эти центры накапливают некоторую обобщенную экономическую информацию с тем, чтобы удовлетворять большинству поступающих запросов без обращения к первичным центрам. Однако в случае необходимости может быть осуществлена автоматическая выдача любой справки из одного первичного центра в другой. Связь при этом осуществляется через соответствующие территориальные центры.

Верхний ярус состоит из одного центрального вычислительного центра, в задачу которого входит диспетчеризация работы всей сети. Кроме того, этот центр играет роль территориального центра для общегосударственных учреждений, расположенных в Москве. Остальные учреждения и предприятия центрального района обслуживаются обычным территориальным центром. Разумеется, роль подобной системы нельзя сводить лишь к выдаче справок для прямого использования их людьми. Основная ее задача состоит в быстрой подготовке необходимых информационных массивов для решения на ЭВМ различного рода задач оптимального планирования и управления. Однако даже в своей чисто справочно-информационной части она представляет собой весьма большую систему, для рационального проектирования которой необходимо иметь специальные приемы.

Каковы же основные принципы, на которых строится проектирование современных систем обработки данных, в том числе и информационно-справочных систем?

Прежде всего, как правило, оказывается не целесообразным ориентировать автоматизированные системы обработки данных на тот круг и объем задач, которые решаются в обычных неавтоматизированных системах. Основной эффект автоматизации обработки данных не просто в экономии человеческого труда, а в возможности решения принципиально новых задач или такого увеличения объема и скорости решения обычных задач, которое приводит к возникновению нового качества. В этом состоит сегодня одна из первых и важнейших аксиом системотехники.

Эта аксиома накладывает отпечаток на порядок организации проектирования автоматизированных систем обработки данных. Обычный способ, начинающийся с изучением потоков входной информации, не учитывает существенного изменения этих потоков при постановке принципиально новых задач, не решавшихся до проведения автоматизации. Поэтому целесообразно изменить подобный порядок, сделав основной упор на первоочередном изучении не столько входной, сколько выходной информации проектируемой системы.

Проектируемую систему целесообразно разбить на четыре основные части. Это, во-первых, техническая база системы переработки информации и оформления выходных документов. Во-вторых,— внутреннее математическое обеспечение системы, включая так называемую службу массивов. В-третьих, это совокупность программ, решающих задачи обработки, на которые собственно и рассчитана проектируемая система. Программы эти включают в себя и программы оформления выходных документов системы. Наконец четвертая часть системы охватывает вопросы подготовки, сбора и передачи входной информации, включая выбор необходимых технических средств и форм первичных документов.

Разработка всех частей системы производится параллельно в несколько этапов, на каждом этапе происходят уточнение и согласование взаимных требований отдельных частей системы. При этом основную координирующую роль играет служба массивов. Чтобы понять смысл этой роли, рассмотрим подробнее начальный этап разработки. Вначале главный

конструктор системы при помощи специальной группы исследователей операций и системотехников проводит предварительное изучение автоматизируемого объекта и намечает основные контуры соответствующей системы обработки данных. На этом этапе производится выработка перечня решаемых системой задач и устанавливается ориентировочный вид выходных документов системы.

Предположим для определенности, что речь идет о системе обработки данных для управления крупным машиностроительным заводом. В таком случае на основании исследования операций и моделирования избранного объекта главный конструктор может установить, скажем, что наиболее важными задачами с точки зрения возможного экономического эффекта являются задачи оперативно-календарного планирования управления запасами и экономического анализа составляющих себестоимости выпускаемых изделий. На основании моделирования может быть установлена (в первом приближении) желательная степень подробности и точности решения указанных задач. В соответствии с этим выбираются предварительные формы выходных документов для этих задач. Как правило, подобные документы оказываются новыми, не существовавшими до внедрения автоматизированной системы.

Специальная группа, ответственная за организационно-технические мероприятия в связи с разработкой и внедрением системы, согласует эти документы с руководством завода и определяет порядок их использования управленческим аппаратом и цехами. Кроме того, формируется список уже имеющихся выходных документов, составление которых передается автоматизированной системе без существенного изменения применяющихся при этом методов обработки. В нашем случае это могут быть различные бухгалтерские документы, сводки о выполнении плана, статистические отчетности и т. п.

Далее производится первичный выбор технических средств и осуществляется предварительное распределение машинного времени для решения задач обработки данных. При этом оставляется не менее чем двукратный запас времени для работы диспетчерской программы и программ корректировки и подготовки массивов. На каждую задачу (оканчивающуюся изготовлением соответствующего входного документа) выделяется ответственный программист или группа программистов. Программисты программируют свои задачи в одном из универсальных алгоритмических языков, обладающих средствами для описания оборудования (например, КОБОЛ). Им предоставляется право предъявлять требования группе службы массивов о необходимых массивах исходных данных и об их расположении на имеющемся оборудовании в момент, когда соответствующая задача вызывается на счет.

Требования к составу, расположению и способу упорядочения массивов у разных программистов могут оказаться (и обычно оказываются) противоречивыми. Например, массив, содержащий сведения об оборудовании завода, для одной задачи целесообразно иметь упорядоченным по типам станков, для другой — по датам их установки, для третьей — по цехам и т. д.

Перед службой массивов встают при этом противоречивые требования. Если удовлетворить запросы всех программистов за счет организации и постоянного хранения в системе всех требуемых массивов, то время решения задач обработки может сократиться. Однако возникает не всегда оправданная избыточность, поскольку одни и те же данные будут многократно повторяться в составе различных массивов. Это приведет к необхо-

димости увеличения оборудования и к резкому росту времени работы программ обновления данных в массивах.

Согласовать возникающие противоречия можно лишь на основании моделирования работы системы. В зависимости от соотношения между частотой использования массива и частотой обновления содержащихся в нем данных, а также в результате учета конкретных особенностей используемого оборудования и внутреннего математического обеспечения используемых ЭВМ может оказаться более выгодным то или иное решение.

Если те или иные данные изменяются относительно редко, а запрашиваются (задачами обработки данных) сравнительно часто, то может оказаться выгодным организовать эти данные в несколько различных массивов (в соответствии с требованиями программистов). Если же подобное решение оказывается невыгодным, то возможны другие два решения. В первом случае в результате взаимных согласований находится такая форма представления массива данных, которая в среднем лучше других удовлетворяет всех программистов, использующих эти данные.

Во втором случае специализированная программа-диспетчер, обслуживающая систему, перед вызовом на счет той или иной задачи подготавливает массив данных в той форме, на которую эта задача рассчитана. С этой целью в состав внутреннего математического обеспечения системы вводятся специальные программы сортировки и формирования массивов. После решения задачи сформированные для ее решения массивы могут быть стерты, тем самым освобождаются соответствующие объемы памяти.

Группа, занимающаяся разработкой алгоритмов и программ для той или иной задачи обработки данных, определяет влияние на результат решения этой задачи задержки времени обновления исходных данных. Учитывая эту зависимость, группа службы массивов на основании моделирования системы устанавливает целесообразную величину указанной задержки. Это означает, что изменения данных не сразу вносятся в соответствующие массивы, а накапливаются в специальных массивах изменений. По достижении известного момента времени или известной величины массива изменений программа-диспетчер прерывает работу системы и вызывает программу, разносящую изменения по соответствующим массивам.

Группа службы массивов вместе с группой входной информации и группой организационно-технических мероприятий решает вопросы, связанные с первичным формированием и последующим обновлением всех используемых в системе массивов. Учитывая необходимую частоту обновления, максимально допустимое время задержки и характер обновляемых данных, выбираем форму входных документов, способ их приготовления и передачи в систему. Если максимально допустимое время задержки мало, то используем устройства, работающие с машиной в реальном масштабе времени (специальные датчики, непосредственно связанные с машиной телетайпы и т. п.). В другом крайнем случае, когда допустимая задержка достаточно велика, иногда возможна доставка документов вручную с последующим их перфорированием и вводом в машину. Однако подобного способа ввода данных желательно по возможности избегать, так как он не только требует дополнительных затрат труда, но и приводит к возникновению дополнительных ошибок.

Определяется также перечень лиц, ответственных за подготовку входных документов, и четко фиксируется круг их обязанностей. Специализированная программа-диспетчер системы следит за правильностью документооборота в системе, осуществляя в случае необходимости соответствующие напоминания.

В процессе работы отдельных групп постоянно возникают те или иные изменения, требующие согласования, желательно немедленного. Подобная координационная работа осуществляется в основном группой службы массивов и группой, разрабатывающей программу-диспетчер. Изменения, о которых идет речь, могут касаться вида используемых массивов данных, их расположения в памяти, времени решения задачи, желательной частоты и времени задержки обновления данных. Эти изменения могут в ряде случаев приводить к необходимости пересмотра ранее принятых решений в отношении технической базы (увеличения или уменьшения числа магнитных лент печатающих устройств, изменения производительности центрального процессора и т. п.).

Руководители групп службы массивов и системного диспетчера осуществляют таким образом постоянный контроль за ходом работы остальных групп. Обычно при разработке достаточно сложных систем необходимость в частичных изменениях возникает столь часто, что, например, частота обращений программистов по поводу уточнений вида используемых массивов или времени задачи может служить известным мерилом интенсивности их работы.

Сложность современных систем обработки данных и соответствующих систем программ делает обязательным использование тех или иных методов автоматизации программирования. Разработке системы обработки данных должна предшествовать разработка систем интерпретации или трансляторов с одного или нескольких алгоритмических языков. Все внешние задачи обработки данных должны, как правило, программироваться в тех или иных проблемно-ориентированных языках с последующей трансляцией или интерпретацией. Задачи внутреннего математического обеспечения (системный диспетчер, программы подготовки, обновления и формирования массивов данных) программируются обычно на машинно-ориентированных языках (типа автокодов).

Как показывает практика, создание систем обработки данных значительно облегчается, если язык машины расширяется специальной системой макрокоманд, предназначенных для работы с массивами данных. Опишем одну систему для работы с массивами, построенную в Институте кибернетики АН УССР. Для каждого из операторов нужно иметь несколько макрокоманд в зависимости от того, помещаются массивы, с которыми они оперируют, в оперативной памяти, на магнитных барабанах, на магнитной ленте или на нескольких магнитных лентах.

Операторы делятся на несколько групп в зависимости от характера их работы. Первую группу составляют так называемые **операторы-выборки** $C_{\text{В}}$.

Оператор групповой выборки $C_{\text{ВГ}}$ работает с двумя массивами M_1 и M_2 и некоторым отношением $P(x, y)$ между признаками x и y элементов этих массивов. Он строит новый массив M_P из всех тех фраз массива M_1, признак x которых находится в отношении P с признаком каких-либо фраз управляющего массива. Например, если M_1 — массив нормативов, M_2 — массив плана выпуска, x и y — шифр изделия, отношение $P(x, y)$ — совпадение шифров $x = y$, то оператор групповой выборки $C_{\text{ВГ}}$ построит массив нормативов лишь для изданий, включенных в план.

Оператор сокращения массива $C_{\text{ВС}}$ в противоположность предыдущему строит массив M_P из всех фраз массива M_1 (за исключением тех фраз, которые удовлетворяют отношению $P(x, y)$).

Оператор подборки $C_{\text{ВП}}$ строит массив, в который после каждой очередной фразы управляющего массива M_2 выписываются все находящиеся с ней в отношении P фразы массива M_1.

Оператор выборки экстремальных фраз $C_{ВЭ}$ строит массив из всех фраз заданного массива, обладающих максимальным (или минимальным) значением заданного признака x.

Оператор анализа массива $C_{ВА}$ производит подборку и подсчет фраз заданного массива, элементы которых удовлетворяют заданному отношению.

Вторую группу составляют так называемые **операторы сортировки** $C_С$.

Оператор упорядочения $C_{СУ}$ переписывает элементы заданного массива в порядке возрастания или убывания заданного признака x.

Оператор группировки $C_{СГ}$ переписывает данный массив таким образом, что все фразы с одинаковым значением заданного признака x записываются рядом.

Оператор слияния массивов $C_{СС}$ из двух заданных массивов M_1 и M_2, одинаково упорядоченных по какому-либо признаку x, строит новый массив M_P, включающий все фразы массивов M_1 и M_2 и упорядоченный так же, как и исходные массивы.

Оператор таблиц $C_{СТ}$, сохраняя последовательность фраз в заданном массиве M_1, переставляет одним и тем же образом элементы этих фраз (столбцы таблицы).

Третью группу составляют **операторы изменения значений памяти** Ф.

Вычислительный оператор по обработке одного массива $Ф_{В1}$. Для фраз данного массива M_1, удовлетворяющих данному условию P, вычисляется заданная функция ф. Результаты этих вычислений вместе с выделенными элементами первоначальных фраз составляют новый массив M_P.

Совершенно аналогично определяется вычислительный оператор $Ф_{В2}$ по обработке двух массивов.

Оператор внесения изменений $Ф_И$. Для каждой пары фраз двух данных массивов, которые удовлетворяют заданному отношению P, производится замена выделенных элементов первой фразы соответствующими элементами второй фразы.

Две последние группы операторов составляют операторы обмена с внешними устройствами и операторы управления, строящиеся в смысле традиционных машинных операций обмена и переходов по ключам.

Статистика, проведенная в Институте кибернетики АН УССР, свидетельствует о том, что трудоемкими являются операторы сортировки. В ста исследованных задачах с большими массивами информации частота применяемости операторов этой группы (среди всех других описанных операторов) составила 35 %, а их выполнение заняло 70 % машинного времени (на машине «Минск-22»). Соответствующие результаты для операторов выборки составили 30 и 10 %, а для операторов изменения значений памяти — 20 и 10 %.

Эти данные свидетельствуют о необходимости уделить самое серьезное внимание вопросам повышения быстродействия ЭВМ при выполнении сортировочных операций. Наиболее эффективными мерами для этой цели являются увеличение емкости оперативных запоминающих устройств, повышение частоты обмена информацией с магнитными лентами, а также включение в систему быстродействующих накопителей на магнитных дисках. Для построения достаточно эффективных систем обработки данных в ближайшие годы нам потребуются ЭВМ с оперативной памятью на 128—256 тыс. слов и магнитные ленты, работающие на частоте порядка 300 кГц.

В отношении вводных устройств отметим, что помимо традиционных флексорайтеров, телетайпов, считывающих перфораторов, читающих автоматов для магнитных чернил и обычных шрифтов за последние годы все

большее значение начинают приобретать экраны со световым пером. В справочно-информационной системе одной из больших американских поликлиник такие экраны используются для ввода больными различного рода данных, необходимых для истории болезни. На экране высвечивается вопрос и множество возможных ответов на него. Пациенту достаточно лишь подчеркнуть световым пером нужный ответ. Подобные экраны в системах с разделением времени могут заметно повысить эффективность многих справочно-информационных систем и систем обработки данных.

Весьма важным фактором повышения производительности систем обработки данных является мультипрограммирование. В ряде случаев разумное использование мультипрограммирования позволяет в известной мере компенсировать недостатки оборудования. Известно, например, как отражается на эффективности справочно-информационных систем отсутствие внешних ЗУ с произвольной выборкой (типа магнитных дисков). Если, однако, производить обработку запросов не поодиночке, а группами, то хотя задержка получения ответа и увеличивается, но общая производительность системы (использующей в качестве накопителей лишь магнитные ленты) может быть сделана достаточно высокой.

Комбинация мультипрограммной обработки с разделением времени между внешними потребителями, необходимость постоянного обновления и формирования массивов сильно усложняют системный диспетчер. Вместе с тем от его качества в значительной мере зависит производительность системы. Разработка эффективного диспетчера, как и общей структуры системы, невозможна без проведения тщательного системного моделирования. При разработке первых систем моделирующие программы создавались вручную и требовали большой затраты труда и времени.

В настоящее время для этой цели используются системы автоматизации программирования, базирующиеся на входных языках, специально ориентированных на задачи моделирования ЭВМ и систем обработки данных. К числу таких языков относятся СОД, СИМСКРИПТ, СЛЭНГ и др. Последний язык разработан в Институте кибернетики АН УССР. Он объединяет в себе большинство положительных качеств, присущих зарубежным языкам моделирования.

В языках моделирования существуют средства для описания параллельно протекающих процессов, процессов прерывания и «захвата» устройств теми или иными сообщениями. Большое место в языке занимают средства для генерирования случайных чисел и последовательностей с различными законами распределения. Для условных переходов широко используются логические условия типа «устройство занято», «память заполнена», «память пуста» и т. п. Ряд языков (СИМСКРИПТ, СЛЭНГ и др.) пользуется списочными структурами и соответствующими операторами.

Использование языков моделирования при наличии соответствующих трансляторов или систем интерпретации позволяет сравнительно быстро и просто моделировать сложные системы обработки данных. Вместе с тем в ряде случаев применение методов Монте-Карло в их чистом виде может быть связано с таким большим машинным временем для моделирования, что исследование большого числа вариантов построения исследуемой системы окажется затруднительным и даже практически невозможным.

Поэтому в последнее время разрабатываются методы комбинированного моделирования, в которых функционирование отдельных участков систем обработки данных описывается аналитическими выражениями (как системы массового обслуживания), а метод Монте-Карло используется для моделирования связей этих участков и системы в целом.

КИБЕРНЕТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА

(В книге «Научно-технический прогресс в УССР 1961—1970 гг.», Киев.— 1970)

Кибернетика и электронная вычислительная техника, представляющие собой составную органическую часть научно-технического прогресса, являются действенной производительной силой социалистического общества.

В любой сфере деятельности человеческого коллектива — в области науки, техники, экономики, производства — методы и средства кибернетики способствуют повышению производительности труда. Это, в свою очередь, дает возможность наиболее эффективно и целесообразно использовать материальные и людские резервы общества, планомерно развивать организацию всего общественного хозяйства.

Научно-технический прогресс не только вызвал к жизни кибернетику и электронную вычислительную технику, но и создал основу для их становления. Многообразие свойств электрона и атома, раскрытых, но еще полностью не изученных наукой, способствовало развитию современной электроники, без которой создание и дальнейшее совершенствование электронных вычислительных машин представлялось бы неразрешимой проблемой. Отметим, что именно ЭВМ и средства электроники составляют техническую базу кибернетики.

Возникновение кибернетики как теоретической основы автоматизации в сфере науки, техники и производства было подготовлено всем ходом развития научно-технического прогресса.

На Украине кибернетика и вычислительная техника достигли первых успехов благодаря большим запросам практики, решению насущных задач народного хозяйства.

Историю развития кибернетики на Украине и становление ее как науки условно можно разбить на несколько этапов.

Первый этап является как бы предысторией развития кибернетики и вычислительной техники, когда были сделаны первые попытки создания электронных вычислительных машин. В этот период проводились прежде всего работы в области аналоговых вычислительных машин (Киевский государственный университет им. Т. Г. Шевченко, Академия наук УССР), но еще не были организованы научные учреждения, которые бы вплотную занимались проблемами создания вычислительной техники.

В 1947 г. в Институте электроники АН УССР была основана лаборатория моделирования и вычислительной техники под руководством академика С. А. Лебедева. Коллектив лаборатории разработал первые устройства дискретной электронной вычислительной техники. Позже здесь была создана первая в Советском Союзе и в континентальной части Европы электронная цифровая вычислительная машина для научных расчетов «МЭСМ» (малая электронная счетная машина). Создание «МЭСМ» явилось знаменательным событием в истории развития вычислительной техники.

Несмотря на то что «МЭСМ» была первенцем, она характеризовалась универсальностью в алгоритмическом отношении, что является одним из основных позитивных качеств электронных вычислительных машин. Машину использовали для большого количества важных расчетов, в частности для Куйбышевской ГЭС и ряда других крупных социалистических строек того времени.

С работами этой лаборатории была связана деятельность таких крупнейших ученых, как А. А. Дородницын, М. В. Келдыш, М. А. Лаврентьев, которые занимались прикладными вычислительными работами, используя лабораторию в качестве базовой для научных исследований.

Зарождаются и другие ячейки по созданию и развитию вычислительной техники. Так, в Институте математики АН УССР П. Ф. Фильчаков успешно занимался созданием моделирующих устройств типа ЭГДА, приобретших впоследствии известность и выпускаемых серийно. В Днепропетровском институте инженеров железнодорожного транспорта под руководством В. А. Лазаряна создавались устройства для моделирования динамики подвижных составов на железных дорогах.

В лаборатории, которой руководил С. А. Лебедев, несмотря на то что основные научные силы перебазировались в Москву для работы над большой электронной вычислительной машиной «БЭСМ», начались работы по созданию специализированной машины «СЭСМ» для решения систем линейных алгебраических уравнений. В дальнейшем лаборатория была преобразована в лабораторию вычислительной техники Института математики АН УССР. В это время партия и правительство указывали на необходимость дальнейшего развития вычислительной техники. Было решено создать ряд республиканских вычислительных центров. Один из них планировалось построить в Киеве; организация его была поручена Академии наук УССР.

Следующий период можно назвать периодом становления кибернетики как науки, периодом создания материальной базы для развития кибернетики и вычислительной техники. Для него характерна организация многих самостоятельных научных учреждений на Украине, в которых впоследствии велись работы по разным научным направлениям.

Согласно постановлению партии и правительства создаются вычислительные центры в Киевском государственном университете им. Т. Г. Шевченко и в некоторых научно-исследовательских учреждениях.

Из состава Института математики АН УССР выделилась лаборатория вычислительной техники, на базе которой в 1957 г. был образован Вычислительный центр Академии наук УССР с правами научно-исследовательской организации.

Одним из основных научных достижений данного этапа явилась завершающая работа по созданию машины «СЭСМ», а также разработка новой ЭВМ «Киев». Созданный образец машины «Киев» стал основой для оснащения самого Вычислительного центра, второй экземпляр машины использовался в Объединенном институте ядерных исследований (Дубна), а впоследствии был передан в Морской гидрофизический институт АН УССР (г. Севастополь).

В машине были применены оригинальные принципы построения схем, в частности центрального управления. С помощью ЭВМ «Киев» впервые в нашей стране осуществлено опытное дистанционное управление производственными процессами. Это было достигнуто вследствие создания регистрирующей цифровой установки, обеспечивающей сбор информации с датчиков, преобразование их показаний в дискретный код и поступление

данных в канал связи. Таким образом, была доказана возможность использования ЭВМ для управления производственными процессами и положено начало созданию автоматизированных систем управления производством. На машине «Киев» проводились также интересные научные эксперименты по распознаванию образов, по комплексированию ЭВМ «Киев» с аналоговой машиной и т. д.

В этот период были сформулированы принципы последующего развития теории вычислительных машин: автономного функционирования устройств ЭВМ, микропрограммного управления, ступенчатой организации управления и т. д.

Отметим, что современное развитие математики тесно связано с появлением и быстрым прогрессом электронных вычислительных машин. В математике складывается новый символический язык алгоритмов. История развития общих алгоритмических языков (в прикладном аспекте) насчитывает около десяти лет.

Большой вклад в развитие новых областей математики внесен украинскими учеными. Закладываются основы алгебры алгоритмических языков, т. е. способов, позволяющих осуществлять формальные преобразования записей в этих языках (впоследствии эта теория была изложена в докладе на Международном математическом конгрессе, 1966 г., Москва).

Возникновение электронных вычислительных машин с программным управлением вызвало к жизни новый прикладной раздел современной теории алгоритмов — программирование. Эта область науки связана с изучением средств автоматического решения задач самых разнообразных видов, в частности возможности общения человека с машиной, которая начинает использоваться в самых различных сферах человеческой деятельности.

Создание электронных вычислительных машин «МЭСМ» и «БЭСМ» с программным управлением положило начало методике программирования во внутреннем языке этих машин с учетом ограниченности их памяти. Из-за недостатка широких практических приложений работы велись в основном по исследованию различных способов программирования.

Исследования киевских математиков связаны с математической разработкой и построением большой по тому времени ЭВМ «Киев». Ученые, участники функционирующего при Институте математики АН УССР, позднее при Вычислительном центре АН УССР, а затем в Институте кибернетики АН УССР семинара, разработали и предложили ряд способов записи алгоритмов и методов программирования — граф-схем, специализированных программирующих программ и адресного программирования.

Метод граф-схем применялся для подготовки математических и логических задач при их решении на электронных вычислительных машинах. Этот метод оказал большое методологическое влияние на развитие теории программирования.

Метод специализированных программирующих программ сыграл большую роль при составлении библиотек стандартных программ для ЭВМ «Киев», «Урал», М-20», а впоследствии при разработке математического обеспечения малых вычислительных машин с развитой системой непосредственной интерпретации типа «Промінь», «МИР».

Киевские математики наибольшее внимание уделили разработкам адресного алгоритмического языка, методам составления трансляторов и их созданию для ряда отечественных машин. Идея построения адресного языка, в основу которого положены понятия косвенной адресации и ранга адреса, была обоснована В. С. Королюком и Е. Л. Ющенко.

Адресный язык стал применяться как входной язык систем автоматизированного программирования для машин типа «Урал», «М-20», «Днепр», «Минск», а также как язык для описания трансляторов. Адресный язык был включен Министерством просвещения УССР в учебные программы школ с соответствующей специализацией. Разработанный в Институте кибернетики АН УССР транслятор с адресного языка для машины «Урал-1» был передан для использования в Чехословакию, Венгрию, ГДР.

Многие монографии и учебные пособия (Б. В. Гнеденко, В. С. Королюк, Е. Л. Ющенко. Элементы программирования, 1961 г.; Л. А. Калужнин, В. С. Королюк. Алгоритмы и математические машины, 1964 г., и др). сыграли большую роль в распространении идей программирования и подготовке кадров программистов.

К данному периоду относится и начало работы по внедрению вычислительной техники в производство. Проводились опыты управления технологическими процессами на расстоянии с помощью электронно-вычислительных машин.

Большое внимание уделялось воспитанию и подготовке кадров. В Киевском государственном университете им. Т. Г. Шевченко и в Киевском политехническом институте была введена специализация по вычислительной технике. Организован специальный техникум, первый в Советском Союзе, готовящий специалистов по вычислительной технике. При Доме научно-технической пропаганды в Киеве начали функционировать курсы, на которых обучались не только инженерно-технические работники киевских предприятий, но и специалисты из других городов Украины. Такая подготовка кадров явилась базой для последующего развития электронной вычислительной техники и широкого внедрения ее в производство.

В этот же период вырисовались основные черты научного стиля работы молодого коллектива Вычислительного центра АН УССР, который можно охарактеризовать двумя принципами. Первый — это единство теории и практики. Ученые старались заниматься такими практическими разработками, которые основывались на большой серьезной теории. Вместе с тем проводились теоретические разработки, которые находили практическое применение в народном хозяйстве. В молодой научной организации происходило становление комплексной тематики, при которой завязывались сложные проблемы, разрабатываемые специалистами самых различных отраслей знаний: в области вычислительной техники, управления технологическими процессами на предприятии и т. д.

Второй принцип, который был положен в основу деятельности украинских ученых, — это принцип единства ближних и дальних целей. В период становления кибернетики возникло много заманчивых идей, например моделирование мозга, моделирование творческих процессов и т. д., что имело важное значение для перспективного развития науки.

Но вместе с тем было ясно, что практические результаты по такого рода работам могут быть получены только через 15—20 лет. С одной стороны, отказ от подобного рода работ намного бы сузил горизонты развития науки. С другой, — нельзя было, если руководствоваться первым тезисом, браться за работы, которые в течение 15—20 лет не дали бы никаких практических результатов. Поэтому свою работу ученые пытались строить так, чтобы наряду с дальней целью, с разработкой больших научных проблем в перспективе достигалась бы и ближняя цель, решались те научные задачи, результат которых непосредственно использовался бы в народном хозяйстве уже сегодня.

К концу этого периода в других городах Украины — Харькове, Днепропетровске — возникают ячейки, связанные с развитием электронной вычислительной техники, прежде всего с эксплуатацией ЭВМ и созданием системы математического обеспечения для решения тех или иных классов инженерных и научных задач с помощью машин.

Годы 1959—1962 можно считать периодом развернутого фронта исследований, началом создания промышленной базы вычислительной техники, выходом украинских ученых-кибернетиков на всесоюзную научную арену. Для этих лет характерно формирование основных научных направлений в области кибернетики, по которым Академия наук УССР заняла ведущее положение в Советском Союзе.

Одно из научных направлений — теория автоматов — начало складываться в начале 50-х годов в результате проектирования вычислительных машин, а также разработки математических моделей высшей нервной деятельности биологических организмов.

В Вычислительном центре АН УССР теория автоматов получила свое дальнейшее развитие. Здесь была создана теория, удобная для практических применений. Научные работы этого времени были посвящены усовершенствованию языка регулярных выражений, построению новых алгоритмов анализа и синтеза автоматов. Алгоритм синтеза был положен в основу этапа абстрактного синтеза при проектировании схем и устройств вычислительных машин. В дальнейшем была разработана полная методика синтеза схем дискретных устройств преобразования информации, охватывающая основные этапы синтеза формальными методами.

По теории автоматов были выпущены в свет основные монографии, впоследствии переведенные на многие языки мира и сыгравшие большую роль в распространении формальных методов синтеза вычислительных и управляющих машин. Была развита алгебраическая теория автоматов, позволившая установить связь практических нужд развития электронной вычислительной техники с решением многих классических задач алгебры, что сразу расширило фронт исследований.

Результаты теории автоматов нашли многочисленные применения в различных практических задачах, имеющих большое значение для народного хозяйства. Так, благодаря теоретическим исследованиям в этой области была построена методика проектирования вычислительных машин и систем, а также решен ряд задач при проектировании конкретных электронных вычислительных машин, например ЭВМ серии «МИР».

Были получены фундаментальные результаты в теории самоорганизующихся систем, распознавания образов. Все эти вопросы широко освещались в научной литературе.

Одной из основных работ этого периода стала разработка управляющей машины широкого назначения, получившей название «Днепр-1»,— первой в Советском Союзе машины такого класса. Утверждение тематики, связанной с машиной «Днепр-1», было сопряжено с большими трудностями. В то время среди подавляющего большинства научных коллективов, занимающихся разработкой вычислительной техники для нужд управления, господствовала идея специализированных машин, причем разрабатывались ламповые машины.

Киевские ученые работали сразу в двух направлениях: создавали полупроводниковую машину и вместе с тем — для управления многими производственными процессами. Была разработана методика проектирования, совмещающая работу по изготовлению машины с созданием системы математического обеспечения, математических программ; иными

словами, подготавливалась почва для ее внедрения. Эта методика, в частности, выразилась в решении такого рода задач: были разработаны сравнительно простые специальные устройства, которые устанавливались на предприятии и с помощью которых осуществлялась передача данных непосредственно с производственных предприятий на большие расстояния, например из Днепродзержинска, Славянска и других городов Украины в Вычислительный центр АН УССР.

Так, машина «Днепр-1» управляла в течение 80 ч химическим процессом в Славянске, в течение 48 ч процессом выплавки стали в Днепродзержинске. ЭВМ «Днепр-1» еще принималась Государственной комиссией, и решался вопрос о ее серийном производстве, а потребители готовились ее приобрести, рынок сбыта был уже создан, поскольку специалисты заводов подсчитали, какой весомый экономический эффект будет от внедрения этой машины на предприятии. Время процесса плавки, например, в Днепродзержинске существенно сокращалось, а по сравнению с работой лучшего диспетчера, ведущего химический процесс в Славянске, управление химическим процессом с помощью ЭВМ было проведено на более высоком уровне. Следовательно, создание машины «Днепр-1» решило одновременно задачи научные и практические.

Работая над созданием машины широкого назначения, ученые Вычислительного центра поддерживали творческие связи с многими институтами Академии наук УССР и других ведомств, например с Институтом проблем литья АН УССР, в котором проводились работы в области цифровых вычислительных устройств, предназначенных для управления различными участками технологического процесса в вагранке. Определенные результаты в этом направлении были получены в Северодонецке, в Вычислительном центре Киевского государственного университета им. Т. Г. Шевченко и других организациях Украины.

Расширились работы, проводимые Вычислительным центром АН УССР совместно с Институтом автоматики (г. Киев), по созданию системы «Авангард» для автоматизации процесса проектирования и изготовления деталей корпуса судов на Николаевском судостроительном заводе.

Система «Авангард» позволила автоматизировать важнейшие и наиболее трудоемкие звенья технологического процесса: развертку деталей корпусов на плоскость, плазовые работы и раскладку деталей. Тем самым была существенно повышена производительность труда, снижена себестоимость производства.

В этот же период развернулись работы по созданию аналоговых вычислительных машин. Были изготовлены «ЭМСС-7», «ЭМСС-8 (электронная модель стержневых систем). В 1961—1962 гг. была создана аналоговая машина «Итератор-1», выпускаемая впоследствии серийно.

В экономической кибернетике в этот период группой ученых под руководством В. С. Михалевича были заложены основы последовательного анализа вариантов, позже названного «киевским веником». Этот метод использовался во многих вычислительных центрах как в нашей стране, так и за рубежом. Проводились совместные работы коллективами Вычислительного центра АН УССР и Госплана УССР, решались интересные транспортные задачи.

В ряде городов при политехнических институтах, университетах появились лаборатории, тематика работ которых связана с кибернетикой. Такие лаборатории были созданы и на заводах, и в научно-исследовательских институтах.

Следующий период (с 1962 г.) ознаменован комплексным подходом к решению проблем создания сложных систем управления, организацией мощной базы кибернетической индустрии на Украине, широким представлением научных работ по кибернетике на международной арене.

В эти годы в Донецке был создан Вычислительный центр, который занимается проблемами прикладной математики. Начали развиваться исследования в новых организациях, например в Морском гидрофизическом институте АН УССР и других учреждениях. Над проблемами кибернетики, такими, например, как создание систем автоматизированного управления производством на базе ЭВМ, работает несколько сотен предприятий и научно-исследовательских организаций Украины.

В феврале 1962 г. Вычислительный центр АН УССР был преобразован в Институте кибернетики АН УССР. Это явилось условием для привлечения в институт широкого круга специалистов и дало возможность создать сектор технической и биологической кибернетики. Продолжались работы в традиционном направлении, развивались математические методы классического численного анализа, хотя в этом отношении не было достигнуто больших успехов, поскольку институт специализировался в основном на разработке математических методов, связанных с управлением производственными процессами и с экономикой, а также с теорией программирования. Ныне Институт кибернетики АН УССР, который за большие успехи в развитии кибернетики и подготовку высококвалифицированных научных кадров для Украины и других братских союзных республик Указом Президиума Верховного Совета СССР в 1969 г. был награжден орденом Ленина, является основной научной базой развития кибернетических исследований на Украине.

За сравнительно короткое время Институт кибернетики АН УССР становится ведущим кибернетическим центром страны. В этот период был принят ряд постановлений ЦК КП Украины и Совета Министров УССР о развертывании промышленной базы по созданию вычислительной техники в Киеве, Северодонецке и ряде других городов, т. е. положено начало созданию кибернетической индустрии на Украине. Создается промышленность вычислительных и управляющих машин. Можно сказать, что свыше 30 % парка электронных вычислительных машин в нашей стране составляют ЭВМ, разработанные в Институте кибернетики АН УССР.

В создании и внедрении в практику электронных вычислительных машин принимают участие коллективы Института автоматики (г. Киев), ряд предприятий и заводов Украины, в частности Северодонецкий приборостроительный завод, где создавались новые машины.

Ученые Киевского государственного университета, некоторые научно-исследовательские организации Днепропетровска занимались разработкой специализированных машин и устройств.

Широко проводились исследования в области вычислительной математики и техники в высших учебных заведениях Украины — в университетах Киева и Харькова, в политехнических институтах Киева, Львова. Этот период характеризуется началом резкого качественного роста кадров. В Институте кибернетики АН УССР было защищено много кандидатских и докторских диссертаций, что создало основу для будущего распространения вычислительной техники на Украине не только в системе Академии наук, но и за ее пределами.

Если проследить за развитием одной из научных разработок института — теорией автоматов, создавшей заслуженную славу коллективу, то увидим следующее. На базе теории автоматов была создана малая

система автоматизации проектирования машин, получившая высокую оценку специалистов. Новая система позволила снизить затраты на проектирование ЭВМ и значительно сократить сроки конструирования тех или иных специализированных дискретных устройств. Например, за две недели (рекордный срок) был спроектирован цифровой дифференциальный анализатор «МИМ».

В 1964 г. работы сотрудников института по теории автоматов были удостоены Ленинской премии. Работы молодых ученых в этой области науки отмечены премией Ленинского комсомола (1968 г.).

На современном этапе коллектив Института кибернетики АН УССР работает над созданием большой системы автоматизации проектирования, которая позволит решить задачу автоматизации проектирования не только специализированных малых блоков ЭВМ, но и всей электронной вычислительной машины в целом.

Завершение разработки первой очереди автоматизированной системы проектирования электронных устройств вошло в социалистические обязательства, принятые коллективом института в честь 100-летия со дня рождения В. И. Ленина. В обязательства были включены также разработка и внедрение в производство и практику инженерных расчетов машины «МИР-2», работы по комплексной автоматизации производства этилбензола на Днепродзержинском химическом комбинате. Выполнение социалистических обязательств явилось новым вкладом в утверждение ленинских идей по развитию науки в стране, по созданию материально-технической базы коммунизма.

Развернулись работы по созданию малых машин для инженерных расчетов серии «МИР». Первая машина под названием «Промінь» была разработана в 1962—1963 гг. Она экспонировалась на Международной выставке в Генуе в 1964 г., на ВДНХ, награждена дипломом первой степени ВДНХ, а разработчики машины — двумя золотыми, одной серебряной и семью бронзовыми медалями.

В 1965 г. была сдана Государственной комиссии и запущена в серийное производство вычислительная машина «МИР-1» (машина для инженерных расчетов). Машина нашла широкое применение во многих учреждениях и организациях. Это была, по сути, первая советская электронно-вычислительная машина с полным математическим обеспечением.

Одной из особенностей ЭВМ «МИР-1» является высокий уровень интерпретаций введенных в машину математических понятий. В отличие от других ЭВМ, где обычно для выполнения вычислений заданная программа в алгоритмическом языке с помощью трансляторов переводится на язык самой машины, с которого осуществляется интерпретация, в машине «МИР-1» внешний язык максимально приближен к внутреннему. Это дает возможность легко и просто вносить исправление в программы в процессе их уточнения, позволяет обеспечить тесное взаимодействие человека с машиной, что способствует ее широкому использованию.

Группе ученых Института кибернетики АН УССР за разработку новых принципов построения структур малых машин для инженерных расчетов и математическое обеспечение к ним в 1968 г. была присуждена Государственная премия.

Новая машина этой серии «МИР-2» имеет более совершенные параметры, чем «МИР-1», и может использоваться в исследовательской работе для логических выкладок, автоматизации доказательства теорем, различных преобразований формул. По уровню внутреннего «интеллекта» эта машина в настоящее время не имеет себе равных.

Качественное отличие ЭВМ «МИР-2» от предшествующих машин этой серии «Промінь», «МИР-1» заключается в том, что она может оперировать символами, а это открывает огромнейшие перспективы, поскольку символ — более широкое, более общее понятие, чем число, и им можно обозначить непосредственно целый объект (вектор, матрица и т. д. — все это обозначается символами). Возможности проведения символьных (аналитических) преобразований в ЭВМ «МИР-2» значительно расширяют сферу ее применения.

К тому же, как известно, формулы аналитического вида дают возможность широко изменять параметры, что позволяет получать оптимальное решение задачи. При проектировании технического объекта с помощью аналитического выражения проще подбирать оптимальные параметры этого объекта.

ЭВМ «МИР-2» снабжена экраном, на который выводится рабочее поле программы. С помощью светового карандаша оператор может вносить различного рода пометки, исправления в программу, не прерывая самого процесса вычисления. Работа с экраном значительно экономит время. Общение человека с машиной становится более удобным и рациональным.

Машину «МИР-2» можно удачно использовать в вычислительных центрах, комплексах ЭВМ различного класса (по быстродействию, проблемной ориентации и т. д.). Она может выполнять логические, аналитические, вычислительные операции. В том же случае, когда ее параметры (скорость быстродействия, объем памяти) окажутся недостаточными при проведении объемных вычислений, задачи будут пересылаться в более мощные ЭВМ. Таким образом, машина явится своеобразным связующим звеном больших ЭВМ комплекса вычислительного центра с пользователями.

Разработана и сдана в серийное производство настольная электронная вычислительная машина «Искра», которая предназначена для выполнения научно-технических и учетно-статистических расчетов в проектно-конструкторских учреждениях, в планово-экономических отделах, на машиносчетных станциях. На базе настольной электронной вычислительной машины «Искра» разработан целый ряд малых настольных машин — «Искра-11», «Искра-12», «Орбита» и др. В машине применена память на магнитострикционных линиях задержки. Это дает значительную экономию оборудования и снижает стоимость машины. Работа была удостоена Республиканской комсомольской премии им. Н. Островского. В 1969 г. Государственная комиссия приняла новую машину этой серии — «Рось», получившую хорошую характеристику. «Рось» по сравнению со своей предшественницей более технологична в производстве, у нее более высокая надежность, меньшая потребляемая мощность, уменьшены вес и габаритные размеры. В области управляющих машин продолжается усовершенствование машины «Днепр-1» и ведутся работы, связанные с ее применением. ЭВМ «Днепр-1» используется для управления процессом выплавки стали в бессемеровских конвертерах на Металлургическом заводе им. Дзержинского, Славянском содовом заводе — для автоматизации технологических процессов цеха карбонизации, а также для управления производством на следующих предприятиях: на Горловском азотнотуковом заводе — производством этилбензола, на Львовском телевизионном заводе — производством по выпуску кинескопов, на Рязанском нефтеперерабатывающем заводе — процессом фракционной разгонки нефти, на Чирчикском электрохимическом заводе — процессом получения азотноводородной смеси для синтеза аммиака, на Старобешевской ГРЭС им. В. И. Ленина — энергетическим блоком котел—турбина—генератор

и др. Кроме того, машина «Днепр-1» используется для обработки данных при сложных научных экспериментах. Везде отмечены высокие экономические показатели от использования вычислительной техники.

К 50-летию Советской власти на Украине была закончена новая управляющая система «Днепр-2», снабженная мощным математическим обеспечением. Система хорошо приспособлена для работы с каналами связи в истинном масштабе времени и с датчиками при управлении технологическими процессами. С помощью системы можно решать также задачи планово-экономического характера. Система «Днепр-2», несмотря на меньшее номинальное быстродействие по сравнению с большими ЭВМ, более производительна при решении целого ряда задач, связанных с большим потоком информации.

Эта система предназначена в качестве основного оборудования для различных систем — при управлении непрерывными технологическими процессами, для решения планово-экономических задач в химической, металлургической, нефтеперерабатывающей, горнорудной, текстильной и других отраслях промышленности. Ее можно использовать для оперативного и диспетчерского управления производством на предприятиях машиностроительной, приборостроительной и других отраслей, в обучающих комплексах крупных учебных заведений, для оперативного управления сетью предприятий бытового обслуживания, строительства, транспорта и т. д.

Система «Днепр-2» может быть установлена в кустовых вычислительных центрах, обслуживающих несколько десятков мелких предприятий.

Широким фронтом развернулись работы по созданию систем математического обеспечения для ЭВМ, разработанных не только Институтом кибернетики АН УССР, но и другими организациями различных министерств. В Институте кибернетики проводятся работы по математическому обеспечению машин типа «Минск» и «БЭСМ-6». Для машин класса «БЭСМ» изготовляется транслятор с языка КОБОЛ.

Использование электронно-вычислительной техники позволяет осуществлять более тонкие и глубокие исследования, связанные с решением задач, компактное описание которых выходит за рамки одного проблемно-ориентированного языка. При обработке экономической информации, например, выполняются сложные вычисления, связанные с исследованием операций, линейным программированием, статистическими предсказаниями, а при проведении научно-инженерных расчетов математический аппарат должен быть удобным для представления различных сообщений, сортировки, редактирования данных и т. д.

Вследствие этого был разработан алгоритмический язык для описания экономических задач АЛГЭК, соединяющий в себе указанные выше свойства.

Разрабатываются языки, предназначенные для описания алгоритмов, связанных с проектированием вычислительных машин и систем типа АЛОС, язык машины «МИР» для инженерных расчетов, универсальный машинно-ориентировочный язык АЛМО, язык СЛЭНГ для описания задач моделирования машин и систем и др.

В 1967 г. авторы работ по математическому обеспечению были удостоены Республиканской комсомольской премии им. Н. Островского.

Развитие теории формальных языков, связанных с вопросами построения трансляторов, рационализацией самого процесса конструирования трансляторов, отражено в научных работах данного периода.

Большую роль по координированию исследований и сотрудничеству отдельных коллективов в области программирования сыграла Первая всесоюзная конференция по программированию «Система автоматизации программирования для ЭВМ», состоявшаяся в Киеве в 1968 г. На конференции присутствовало свыше 1500 делегатов из 85 городов страны. Целью конференции было подведение итогов научных работ и определение путей дальнейшего развития в теории программирования и практике построения систем математического обеспечения ЭВМ, координация исследований ученых, работающих в области программирования, ликвидация разрыва между теоретическими исследованиями и разработкой и внедрением результатов в практику.

На конференции обсуждались вопросы теории программирования, построения вычислительных систем, систем программирования, техники трансляции, операционных систем, программирования для специальных применений, разработки систем математического обеспечения, создания специализированных языков программирования и т. д.

Разработка и внедрение систем программирования представляют собой комплексную проблему, связанную в равной мере с решением математических, лингвистических, инженерных и административных вопросов. Ее частичное решение осуществляется на данном этапе развития кибернетики.

Эффективным методом внедрения математического обеспечения в практику работы вычислительных центров, призванных обслуживать различные отрасли народного хозяйства, является создание сети государственных фондов алгоритмов и программ, создаваемых повсеместно в республиках страны.

Украинский республиканский фонд по праву считается пионером в сети государственных фондов. С его помощью были организованы подобные фонды в Грузинской, Азербайджанской, Узбекской, Казахской, Белорусской, Молдавской, Литовской республиках.

Использование алгоритмов и программ фонда дает возможность увеличить полезное время работы ЭВМ.

В области аналоговых машин была развита теория квазианалогового моделирования. На ее основе создан класс новых ЭВМ, получивших путевку в жизнь целыми сериями: «Альфа», «Итератор», «Оптимум», «Аркус», «Асор-1» и «Асор-2».

Специализированные квазианалоговые машины применяются при решении задач прикладной теории упругости, строительной механики, теплотехники, гидромеханики, линейного программирования, сетевого планирования и управления, автоматического регулирования и управления.

Учеными и конструкторами разработана специализированная моделирующая машина для расчетов электрозащиты подземных сооружений от коррозии, возникающей из-за блуждающих токов.

Учеными Института кибернетики АН УССР и Киевского института инженеров гражданской авиации создана серия квазианалоговых счетно-электронных интеграторов для решения задач теории упругости.

Машины демонстрировались на международных выставках в Лейпциге, Брно, Загребе, награждены медалями ВДНХ.

Для управления ионными технологическими процессами создана машина «Киев-67». Она может управлять электронным или ионным лучом, что важно для автоматизации изготовления интегральных схем. Это новейшая технология, которая определяет будущее электроники и электронно-вычислительной техники.

Машина в несколько раз повышает производительность труда при производстве диодных матриц из кремния и обладает удобной системой управления.

В настоящее время в Институте кибернетики АН УССР проводится работа по объединению автоматизированной системы проектирования вычислительных машин с системой автоматизации изготовления элементов и схем, которая основывается на электронной вычислительной машине «Киев-67», с целью полной автоматизации производства ЭВМ.

В этот период создаются принципиально новые автоматизированные системы управления производством в целом.

В 1967 г. была закончена и сдана Государственной комиссии первая очередь автоматизированной системы управления на Львовском телевизионном заводе.

В процессе создания решалась сложная задача организации внутренних систем математического обеспечения (в частности, львовской системы), содержащего 65 тыс. приказов для первой очереди, а с введением в 1968 г. второй очереди — до 150 тыс., т. е. это одна из самых сложных программ, когда-либо составлявшихся и вводившихся в действие в нашей стране для систем управления.

Системе «Львов» придается огромное значение. Это первая система, рекомендованная для серийного производства, в ней комплексно решены вопросы создания необходимой технической базы для многих систем, разрабатываемых в других организациях (в Москве, Ленинграде, Минске и других городах). Внедрение автоматизированной системы «Львов» повышает эффективность использования капитальных вложений в три раза. Вопрос создания комплексной технической базы для автоматизированных систем управления может успешно решаться в организациях типа Института кибернетики АН УССР, где наряду со специалистами в области экономической кибернетики работают крупные специалисты в области технических средств. Такое единение научных сил в настоящее время дает желаемые результаты.

В системе «Львов» имеется такой комплекс средств, который нужен непосредственно для серийного производства и массового внедрения на приборостроительных заводах, выпускающих массовую продукцию. Поэтому она рекомендована для внедрения как типовая система на ряде заводов радиопромышленности СССР.

В процессе создания системы «Львов» решались и практические задачи по организации коллективов системотехников, что было очень трудным и сложным делом, поскольку в стране не было кадров с такой специализацией. В разработке такого рода систем участвовали различные специалисты. Возникали определенные трудности. Специалист, хорошо знающий способы кодирования в каналах связи при передаче данных, может плохо понимать соответствующую методику решения оперативной задачи календарного планирования, которой занимается математик. Сегодня нужны специалисты широкого профиля, хорошо знающие теорию и принципы организации переработки информации в различных видах. Системотехника — это новая отрасль в науке, и подготовка кадров в этой области — задача первостепенной важности.

Институт кибернетики АН УССР явился пионером в разработке и внедрении сетевых методов планирования и управления. Впервые в Киеве на практике осуществлялось комплексное решение вопросов управления и планирования по сетевому методу и управление на базе сетевых графиков реальными строительными объектами. В частности, стро-

ительство моста «Метро» в Киеве велось с помощью сетевых методов, и управление осуществлялось через Вычислительный центр института. Естественно, работа велась совместно со строителями. Строители разработали комплекс мероприятий, позволивший на два месяца раньше срока окончить работы и сэкономить 178 тыс. р. Причем работа была выполнена досрочно без привлечения дополнительных материальных средств.

Научно-исследовательский институт строительного производства и Институт кибернетики АН УССР совместно разработали методику и составили программу расчета оптимальной очередности сроков строительства непрерывным потоком всех крупных тепловых электростанций УССР на двадцатилетие (1963—1983 гг.) — так называемая Южная энергосистема.

Расчет, проведенный с помощью электронно-вычислительной машины, показал, что оптимальный вариант строительства непрерывным потоком позволит получить большой экономический эффект.

Строительство комплекса для получения карбомида на Северодонецком химическом комбинате, которое управлялось также из Вычислительного центра института, было закончено за 18 месяцев вместо 24. По сравнению с проектным заданием стоимость строительно-монтажных работ была снижена на 10,3 %.

Таких примеров можно привести много. В настоящее время молодая организация — Институт автоматизированных систем планирования и управления в строительстве Госстроя УССР — ведет работы по управлению сотнями строек Украины.

В перечень работ украинских ученых входит и участие в строительстве газопровода Средняя Азия — Центр, где применялись новейшие методы динамического программирования, и решение вопроса о распределении нефтепродуктов в масштабе Советского Союза, и решение транспортных задач на Украине, причем комплексных, касающихся не только автомобильного, железнодорожного, но и водного транспорта.

Основные работы по использованию методов кибернетики на транспорте заключаются не только в выполнении отдельных расчетов с помощью ЭВМ, но в основном в разработке типовых информационно-планирующих систем на магистральном и промышленном железнодорожном транспорте. Решение ряда теоретических вопросов по созданию планирующих и управляющих систем на транспорте приобретает особенно важное значение в период ввода в эксплуатацию вычислительных центров на крупных транспортных узлах, например на Донецкой, Приднепровской, Южной, Юго-Западной железных дорогах. Создаются специальные коллективы по изучению возможностей применения ЭВМ в практике транспортных работ.

Работы по использованию вычислительной техники для промышленного транспорта ведутся в пределах республики в таких организациях, как УГПИ Металлургавтоматика (г. Днепропетровск), ВНИИОчермет (г. Харьков), Днепропетровский институт инженеров транспорта и др.

Утверждению экономической кибернетики способствовал тот факт, что сама социалистическая система централизованного планирования и управления народным хозяйством создает наиболее благоприятные условия для развития и внедрения методов экономической кибернетики с использованием современных математических и вычислительных методов. Использование этих методов дает возможность находить оптимальные решения в различных областях экономики и в масштабе народного хозяйства в целом.

Научные исследования в этой области ведутся совместно с Госпланом УССР, министерствами, проектными организациями и передовыми предприятиями Украины.

Госплан УССР и его Экономический институт совместно с Институтом кибернетики АН УССР разработали научную методику классификации промышленной и сельхозпродукции, утвержденную Госпланом СССР как общесоюзную для составления единого классификатора.

Общий экономический эффект от внедрения разработок Института кибернетики АН УССР в народное хозяйство превышает ассигнования, выделенные государством институту по всем статьям расхода.

По ряду работ общий экономический эффект невозможно подсчитать, поскольку институт проводил работы, связанные не только с автоматизацией промышленности, но и с автоматизацией научных экспериментов, например создание автоматизированной системы обработки экспериментальных данных на научно-исследовательском судне «Михаил Ломоносов». Если раньше по возвращении судна из рейса на обработку результатов экспериментальных работ уходило два года, то с внедрением системы вся эта работа проделывалась сотрудниками Морского гидрофизического института АН УССР во время рейса. Методы обычной оценки экономического эффекта тут не приемлемы. Подсчитать зарплату сотрудников, обрабатывающих два года результаты научных исследований, можно, а вот в каких цифрах выразить тот факт, что время научного открытия значительно отодвигается?

Можно привести еще пример. Вводится система автоматизации биологических экспериментов в Киевском научно-исследовательском институте экспериментальной и клинической онкологии. Такая система позволяет вводить данные непосредственно с экспериментальных установок в вычислительную машину. Эти данные тут же обрабатываются, и машина выдает законченные результаты исследований. Во много раз повышается производительность соответствующих экспериментальных установок, но и в этом случае невозможно оценить экономический эффект, который выражается в ускорении развития науки в соответствующей области.

Сейчас ведутся работы в области программирования тех или иных методов автоматического доказательства теорем. Создаются программы для ЭВМ, с помощью которых математик сможет работать вместе с машиной в поисках доказательства новых теорем. Опять-таки темпы ускорения научного прогресса за счет применения электронно-вычислительной техники в отдельных областях науки с большим трудом поддаются экономической оценке.

На данном этапе развития вычислительной техники решаются принципиально новые задачи, в частности закладываются научные основы для создания комплексов разнородных машин, что чрезвычайно важно для стыковки отраслевых систем управления, общегосударственных систем управления с низовыми системами на предприятиях, создающимися в настоящее время. При этом наблюдаются чисто технические трудности. Речь идет не столько о том, чтобы решить задачу экономического планирования конкретно, сколько о том, как с меньшей потерей времени наилучшим образом использовать возможности, заложенные в машинах разных классов.

Ученые создают машины со структурной интерпретацией сложных входных языков. Уже упоминалась машина для инженерных расчетов класса «МИР», при разработке которой выяснилась принципиальная возможность подобных работ для больших машин, что ранее казалось вообще

невозможным. Принципы интерпретации оказались жизнеспособными и стойкими. Более того, они способствовали дальнейшему развитию структуры ЭВМ. Появилась возможность увеличить «интеллектуальность», вложенную в структуру машины, что позволяет сделать скачок вперед по созданию трансляционных систем и приблизить системы программирования к обычным человеческим языкам.

Подготовка математического обеспечения ЭВМ и теория организации вычислительных процессов за последние годы усовершенствовались благодаря работам Института кибернетики АН УССР по созданию собственных машин, а также работам по математическому обеспечению машин, созданных другими организациями.

Ведутся работы по интегральным схемам и микроэлектронике. Диодные линейки, диодные матрицы, разработанные институтом, внедрены в серийное производство.

Были достигнуты определенные результаты и в области биологической и медицинской кибернетики. В Киеве организуется справочно-информационная медицинская система — база для создания медицинского научно-информационного центра, позволяющего решать проблемы эпидемиологии, диагностики и прогнозирования заболеваний сердечно-сосудистой системы на основе изучения данных, полученных от тысячи больных.

Вместе с тем такой центр будет способствовать решению многих проблем, связанных с изучением патогенеза на основе комплексного моделирования физиологических и патологических процессов в человеческом организме. Создана модель функционирования сердечно-сосудистой системы, регулирования углеводного обмена.

В настоящее время идея создания медицинского информационного центра объединяет усилия научных работников Института кибернетики АН УССР и научно-исследовательских институтов Министерства здравоохранения УССР, таких как Институт клинической медицины им. Н. Д. Стражеско, Институт туберкулеза и грудной хирургии им. Ф. Г. Яновского, Институт эндокринологии, Киевский медицинский институт им. А. А. Богомольца, Институт нейрохирургии, базовый санаторий им. М. Ю. Лермонтова в Одессе и др.

Нашли практическое применение ряд устройств, сконструированных в Институте кибернетики АН УССР, например биоэлектрический стимулятор для управления двигательными функциями человеческого организма (для лечебных целей, для управления при профессиональном обучении с целью приобретения навыков, для управления двигательными актами в специальных условиях работы и т. д.). Устройство успешно прошло испытание и используется для лечения в санаториях Евпатории и других лечебных заведениях.

Метод программного управления двигательными функциями разрабатывается и используется в контакте с Министерством здравоохранения УССР (кафедра нервных болезней Киевского мединститута и Клиника нервных болезней им. Октябрьской революции), Республиканским советом по управлению курортами профсоюзов УССР, Институтом медико-биологических проблем Министерства здравоохранения СССР.

В области нейробионики раарабатывается тема по изучению процессов управления и переработки информации в нервной системе человека и животных, рассматривается построение математических моделей бионических устройств, использующих изученные закономерности переработки информации мозгом. Разрабатываются методы управления функциями

центральной нервной системы посредством физических факторов (гибридные биологические системы).

Достаточно широко представлены на Украине работы по изучению организации мозга на уровне элемента (нейрона), блока (рецептивного поля, нервного узла, ядра и т. п.), системы (зрительный, слуховой, вестибулярный анализаторы) и совокупности систем.

Получены математические описания нервного и мышечного возбуждения, методом моделирования изучены ритмические свойства нейрона и блока нервных сетей, разработаны методы изучения структурной и функциональной организации нервных сетей, создана аппаратура для многоканального исследования нервных узлов.

Сюда же примыкают работы, проводимые в Днепропетровском государственном университете, по применению методов моделирования при изучении нейронной дуги спинного мозга.

В Институте физиологии АН УССР проводятся работы по применению электронно-вычислительных машин для исследования процессов передачи и переработки информации в центральной нервной системе. Разрабатываются и применяются математические методы и специальная аппаратура для исследования непрерывных и дискретных характеристик биоэлектрической активности мозга и отдельных его нейронов с использованием ЭВМ.

Интересные результаты получены в области гидробионики, где также проводился ряд исследований.

Институт кибернетики АН УССР совместно с Институтом гидробиологии АН УССР, Институтом гидромеханики АН УССР и другими учреждениями проводит работы по изучению гидродинамики дельфинов, рыб и различных водных животных. Математическое моделирование и машинный эксперимент позволили выяснить механизм больших скоростей движения водных животных и найти способы их физического моделирования. Ведется ряд других исследований в этой области.

Одной из важных задач биокибернетики является комплексное моделирование функций мозга. Модель человеческого мозга даст возможность раскрыть тайны различных творческих процессов человеческого мышления. Создание искусственного мозга — это глобальная проблема, решение которой будет осуществляться поэтапно.

Дальнейшее развитие и обогащение новыми научными идеями получила теория автоматического управления и регулирования, возникшая ранее самой кибернетики, явившаяся для нее вместе с другими областями науки основой становления.

Большую работу в области автоматического регулирования проводит Институт автоматики и многие другие научные коллективы республики. В области систем автоматического регулирования и систем телемеханики получили развитие теория инвариантности и теория комбинированных систем регулирования, на основе которых выполнены конкретные работы.

Разрабатываются сложные системы автоматического регулирования с применением цифровых управляющих машин.

В области самоорганизующихся систем управления получены результаты по решению задач произвольного распознавания образов и ситуаций.

В последнее время на Украине начали широко внедряться вычислительная техника и автоматизация управления производством с помощью ЭВМ. Созданный в Академии наук УССР Научный совет по кибернетике координирует работы различных организаций Украины в области кибер-

нетики и вычислительной техники. Если раньше, например, теорией автоматов занимались только в Киеве, то теперь подобные работы ведутся в Харькове, Львове, Донецке, Ужгороде, Севастополе и других городах страны.

Научный совет по кибернетике организовал свыше 50 семинаров, многие из которых получили всесоюзную и всемирную известность. Например, на семинаре по теории автоматов выступали такие видные зарубежные ученые, как проф. Маккарти из Стенфордского университета (США), проф. Тилле из ГДР, проф. Мацей Столярский из Польши, проф. Кальмар из Венгрии и др.

Институт кибернетики АН УССР достойно представляет советскую науку за рубежом. По линии международных связей представители института входят в Международную федерацию по обработке информации (ИФИП), Международную федерацию по автоматическому управлению (ИФАК), Международную ассоциацию по аналоговым вычислениям (АИКА), комиссии Совета Экономической Взаимопомощи и др.

В 1962 г. на II Конгрессе ИФИП в Мюнхене представитель института сделал доклад по теории самоорганизующихся систем и теории автоматов. Его ввели в состав программного комитета последующего конгресса и предоставили право возглавить подкомитет по теории автоматов, самоорганизующихся систем и искусственного интеллекта.

С этих пор работы по подготовке и разработке программ международных конгрессов по вычислительной технике в области теории вычислительных машин и их применения в научных и инженерных расчетах возглавил коллектив института. В число таких конгрессов входят и III Международный конгресс ИФИП в Нью-Йорке (1965 г.), и IV — в Эдинбурге (1968 г.), и V конгресс, который состоялся в Любляне в 1971 г.

Ученые Института кибернетики АН УССР с 1965 г. не только курируют в ИФИПе работы по моделированию мозга и автоматизации проектирования ЭВМ, но и координируют почти все области приложения электронно-вычислительной техники: естественные науки, инженерно-конструкторские расчеты системы автоматизации конструкторских работ и т. д. Представитель института на IV Международном конгрессе ИФИП-68 был избран председателем программного комитета.

О международном авторитете украинских ученых свидетельствует тот факт, что в ряде стран (США, Англия, ГДР, Венгрия, Югославия, Чехословакия и др.) переведены монографии, написанные учеными Института кибернетики и других научно-исследовательских организаций Украины. В США стали регулярно публиковаться переводы статей ученых Украины. Журнал «Кибернетика» полностью переиздается в США на английском языке, переводится на английский язык журнал «Автоматика».

Большую популярность за рубежом получили труды республиканских семинаров Научного совета по кибернетике, издаваемые Институтом кибернетики АН УССР.

Зарубежные ученые и специалисты высоко оценивают теоретические и многие практические разработки, проводимые в институте (особенно следует отметить высокую оценку, данную английскими и американскими представителями разработанной в Институте кибернетики ЭВМ «МИР»).

Ученые института приняли активное участие в Международной конференции «Наука и общество» (Югославия), в I Международном симпозиуме по биокибернетике (ФРГ), VI Международном конгрессе по медицинской электронике (Швеция), V Международном конгрессе по кибернетике (Бельгия).

В Вашингтоне (октябрь 1967 г.) на I конференции Товарищества кибернетиков с докладом «Моделирование процессов психики» успешно выступил Н. М. Амосов, в Бельгии (г. Льеж) на Международном коллоквиуме по методам оптимизации В. С. Михалевич прочитал доклад на тему: «Последовательный поиск и оптимизация», в Италии (г. Рим) на Конгрессе Международной федерации по обработке информации А. А. Стогний доложил «О проектировании информационно-справочных систем фактографического типа» и т. д.

Сотрудники Института кибернетики АН УССР приглашены для участия в редколлегии четырех зарубежных журналов, публикующих материалы по вычислительной математике, вычислительной технике и кибернетике. Два из этих журналов издаются в США и по одному — в Италии и ГДР.

Особенно тесные связи установились у киевских кибернетиков с представителями ряда стран социалистического содружества. Институт проводил совместные работы с Венгерской академией наук по темам: «Теория цифровых математических машин и автоматизация их проектирования» и «Распознавание образов и читающие автоматы», с Кубинской академией наук по темам: «Исследование процессов производства сахарной промышленности» и «Программирование на счетно-вычислительных машинах». По линии Совета Экономической Взаимопомощи в институте проводится работа по теме «Разработка формальных методов проектирования схем цифровых автоматов и вычислительных машин». С чехословацкими учеными выполнена совместная работа по использованию разработанной и выпускаемой на Украине ЭВМ «Днепр-1» для управления агрегатами тепловых электростанций.

Ученые Института кибернетики принимали участие в консультации болгарского правительства по созданию системы вычислительных центров страны. Работа получила высокую оценку.

Научные работы ученых-кибернетиков Украины вышли на широкую международную арену. Международные связи с каждым годом становятся все шире и прочнее.

С целью дальнейшей концентрации научных сил, обеспечения широкого развития исследований в области кибернетики и автоматизации работ по внедрению электронно-вычислительной техники в народное хозяйство в столице Украины создается Кибернетический центр Академии наук УССР. В нем получат дальнейшее развитие основные направления молодой науки.

За годы развития кибернетики и вычислительной техники возникли новые направления в процессе подготовки кадров.

В Киеве проводится работа по развертыванию подготовки специалистов в рамках новых специальностей. Большую инициативу в данном вопросе проявил коллектив Киевского государственного университета им. Т. Г. Шевченко. В университете, помимо традиционной специализации по вычислительной математике и программированию, существовала также специализация по теоретической кибернетике, экономической кибернетике, математической лингвистике и т. д. Постепенно созрели объективные условия для того, чтобы объединить все эти специальности в одном факультете кибернетики, недавно созданном в университете. Появились реальные возможности не только увеличить количество специалистов, подготавливаемых для нужд народного хозяйства страны, но повысить качество обучения студентов. Есть реальная возможность для проведения работы на этом факультете в тесном контакте с учеными Института

кибернетики АН УССР, а в дальнейшем с коллективом сотрудников Кибернетического центра АН УССР, в состав которого включено строительство специального учебного корпуса, где студентам будет читаться курс лекций по кибернетике учеными не только Украины, союзных республик но и зарубежных стран.

Установлены тесные связи между Институтом кибернетики и Московским физико-техническим институтом, который готовит кадры для передовых отраслей промышленности. Институт кибернетики является базовым институтом для подготовки кадров в области больших кибернетических систем управления. В институте функционирует кафедра теоретической кибернетики и методов оптимального управления Московского физико-технического института, занимающаяся подготовкой кадров для молодой науки. Молодые специалисты пополняют ряды ученых, принимают научную эстафету от старшего поколения.

В условиях социализма нет социальных препятствий для создания автоматизированных систем управления экономикой в национальных и даже межнациональных масштабах. Техническую базу такой системы будет составлять сеть вычислительных центров, соединенных между собой каналами связи, подобно сети энергетической системы, раскинувшейся по всей территории Советского Союза, идеи и пути создания которой были внесены В. И. Лениным и величественные планы построения коммунизма.

Если суммарная мощность электростанций и других силовых установок определяет энергетическую мощь государства, то мощность вычислительных центров станет показателем его информационно-интеллектуальной мощи. Промышленно-экономический потенциал государства по мере развития научно-технического прогресса все больше будет определяться информационно-интеллектуальной мощью, поскольку только высокий уровень информационной вооруженности способствует рациональному использованию производственных и людских ресурсов.

Строящаяся база коммунистического общества будет иметь наиболее эффективную, максимально автоматизированную систему управления экономикой, самые совершенные формы автоматизации производства, к созданию которых прилагают все усилия научные и производственные коллективы страны.

ДИАЛОГ С ВЫЧИСЛИТЕЛЬНОЙ МАШИНОЙ: СОВРЕМЕННЫЕ ВОЗМОЖНОСТИ И ПЕРСПЕКТИВЫ

(Управляющие системы и машины.— 1974.— № 1)

Решение многих задач на ЭВМ связано со значительными трудностями, заключающимися в том, что пользователь не знает заранее рационального пути получения результата, а просчет всех допустимых вариантов привел бы к практически неосуществимым объемам вычислений. Эти трудности имеют принципиальный характер, и их преодоление в настоящее время возможно лишь на основе диалогового взаимодействия человека и ЭВМ.

Диалоговая форма взаимодействия человека с машиной возникла сравнительно недавно и сразу же завоевала заметное место среди различных форм организации вычислительного процесса. Основной идеей в этом взаимодействии является рациональное распределение функций между человеком и машиной на основе взаимного дополнения и использования тех положительных качеств, которыми обладает каждый из партнеров.

Как правило, машина выполняет алгоритмы (или фрагменты алгоритмов), сконструированные человеком, обеспечивает некоторые виды контроля их правильности, а также хранит программы решения отдельных задач и хорошо определенных классов задач.

Управление ходом решения задачи, уточнение ее условия, выбор метода и конструирование алгоритма решения осуществляет человек. Эффект от применения ЭВМ тем больше, чем выше трудоемкость исполнения алгоритма по отношению к его конструированию. При этом для поддержания диалога должна обеспечиваться, с одной стороны, возможность получения ответа машины (результатов решения) через время, не нарушающее процесс мышления пользователя за терминалом, а с другой стороны,— возможность для человека формулировать свои сообщения в лаконичной и компактной форме.

Основным качеством человека, которое еще долгое время будет недоступным для машины, является умение легко ориентироваться в условиях неполной определенности постановки задачи, кажущейся многозначности ее решений и их практической несопоставимости друг с другом. Машина же имеет неоспоримые преимущества при выполнении четко запрограммированных математических вычислений и просмотре логических вариантов.

Оба эти качества существенно необходимы при решении определенных классов задач, объектом анализа в которых является очень сложная, не обозреваемая во всех подробностях структура. Далее приведены примеры таких задач и изложены основные причины, побуждающие решать их в режиме диалога человека с машиной.

Необходимость в диалоговых приемах работы с ЭВМ была отмечена еще на ранних этапах развития вычислительной техники при решении обычных вычислительных задач из научно-технической области. Уже тогда нередко поражало несоответствие между возможностями человека по прочтению и анализу числовой информации и тем огромным количеством цифр, которые, как правило, выводились на цифропечатающее устройство

в результате решения задачи. Причина такого несоответствия очевидна: пользователь ЭВМ практически не в состоянии настолько хорошо проанализировать структуру функций, входящих в условия задачи, чтобы сколько-нибудь точно локализовать область интересных для него значений результатов. Поэтому, не имея прямой связи с ЭВМ и возможности следить за ходом решения, пользователь программировал распечатку результатов в большой области, заведомо перекрывающей прагматически значимую информацию. Выход из этого положения в большинстве случаев тоже очевиден: человек, хорошо знающий условия задачи и возможные связи между переменными, должен получить возможность следить за некоторыми (выбранными вначале наугад) вариантами решения задачи. Анализируя в уме эти варианты, он может с первых же шагов сократить их число и диапазоны изменения переменных и тем самым значительно уменьшить выводимую бесполезную информацию.

Вместе с тем стало ясно, что такая помощь человека (и экономия машинных ресурсов) возможна лишь в том случае, когда результаты пробных вариантов будут представлены человеку в предельно наглядной форме и его вмешательство в вычислительный процесс будет максимально облегчено.

Естественно, что уменьшение числа напечатанных символов не является главной целью введения режима диалога. Как отмечено выше, некоторые задачи (в том числе сложные вычислительные) практически не могут быть решены без участия в процессе решения человека, анализирующего промежуточные результаты и выдающего ЭВМ директивы, управляющие дальнейшим ходом решения. Возникает вопрос: а нельзя ли промежуточные результаты выводить (в малом объеме) на печатающее устройство, затем подвергать их анализу и вводить при следующем сеансе коррективы в ход решения?

Отвечая на этот вопрос положительно, нельзя не учитывать психологические особенности человека: слишком большие перерывы в решении какой-либо задачи неизбежно нарушают ход мыслей, идеи по управлению процессом вычислений забываются, и их приходится восстанавливать, делая определенный шаг назад. В то же время в ряде случаев перерывы в работе помогают найти новую идею и принять правильное решение только «на свежую голову». Это не противоречит режиму диалога, который естественно распадается на сеансы, т. е. представляет собой чередование умственной работы и отдыха, совмещенного с подсознательным вынашиванием идей.

Некоторые общие требования к процессу диалога приведены в [1]. Одна из диалоговых систем, ориентированных на вычислительные задачи, описана в [2].

Типичным является применение диалога в процессе отладки сложных программ. Невозможность предусмотреть заранее все необходимые варианты работы программы для ее полной проверки очевидна, так как уже сам факт необходимости отладки свидетельствует о неполном знании программистом своей программы. Вместе с тем проигрывание всех вариантов для сложных программ невозможно по вполне понятным причинам. В процессе диалога программист быстро «нащупывает» ошибочные участки программы, что делает процесс отладки весьма эффективным. Одна из систем диалоговой отладки в языке АЛГОЛ разработана в Институте кибернетики АН УССР [3].

Человек превосходит и еще долго будет превосходить машины не только в быстроте и адекватности оценки ситуации, но и в выборе общей идеи или генерального плана решения сложной задачи. В этой связи представляется, что разработка полностью автоматических систем для доказательства теорем обречена на неудачу. Высокой оценки в этой области заслуживают

работы Хао Вапга и Н. А. Шапина, однако они, на наш взгляд, важны скорее для развития математической логики вообще, чем для исследования проблемы автоматизации доказательства теорем в прикладном плане. Вместе с тем диалоговая система, в которой человек указывает машине путь поиска доказательства,— это вполне реальная и практически полезная система.

Рассмотрим проблему автоматического (искусственного) перевода. На первых порах, когда эксперименты проводились на ограниченном словаре, казалось, что создание автоматического переводчика — дело техники и может быть осуществлено в ближайшем будущем.

Но естественный человеческий язык — это очень сложное образование, его структуру нельзя вывести из небольшого числа аксиом. Поэтому, первые удачи в автоматическом переводе оказались на многие годы отдаленными от окончательного решения задачи. Реальные успехи на этом пути лежат, на наш взгляд, в области создания человеко-машинного переводчика: машина готовит подстрочник (с альтернативами некоторых слов и выражений, с включением в подстрочник фрагментов первоначального текста, с которого ведется перевод, и т. п.), а человек (специалист в отрасли знаний, к которой относится переводимый текст) доводит этот подстрочник до окончательного варианта, внося в него смысловое единство, устраняя абсурдные выражения, которые могут появиться из-за неопределенности естественного языка. Практическая важность такой диалоговой системы несомненна, и она может быть реализована уже сейчас.

Несколько слов об автоматизации проектирования. Диалог здесь нужен прежде всего для оценки человеком различных вариантов проекта, когда в оценку каким-то образом вводится не только функциональная адекватность проекта, но и его «целостное видение» человеком, эстетические, социальные и эмоциональные факторы, для которых пока не существует формализованных категорий или которые слишком сложно формализовать. Так, при планировке квартиры машине можно поручить задачу размещения жилых и вспомогательных помещений (с какими-либо оптимизационными критериями), однако определение удобства квартиры целесообразно оставить за человеком. Определение удобства является задачей, относящейся к классу задач распознавания образов. По-видимому, многие задачи этого класса еще 10—20 лет будут решаться человеком быстрее и лучше, чем машиной.

Система ПРОЕКТ, разработанная в Институте кибернетики АН УССР, позволяет в диалоговом режиме конструировать и проигрывать различные логические узлы ЭВМ [4].

Аналогичная ситуация возникает при автоматизации шахматной игры, где общая оценка позиции производится шахматистом значительно быстрее и эффективнее, чем ЭВМ. Вместе с тем комбинаторика и просчет отдельных вариантов на некоторое число ходов уже сейчас машиной выполняются лучше, чем классным игроком. Диалоговая система «шахматист ЭВМ» может стать опасным соперником даже для выдающихся шахматистов и послужит важным экспериментальным инструментом не только для развития этой древней игры, но и для исследования процессов совместного решения задач человеком и вычислительной машиной.

Одним из классов задач, требующих диалогового режима решения и имеющих чрезвычайно важное экономическое значение, являются балансовые задачи планирования. Приведем факторы, определяющие сложность этих задач: большой перечень выпускаемых продуктов (десятки тысяч наименований); сложные взаимосвязи между количеством выпускаемых и количеством потребляемых продуктов; наличие неопределенности в граничных

условиях, связанных с наличными людскими ресурсами, потребностями в данном виде продукта и т. д.

Существует много методов формализации этих задач для их последующего решения на ЭВМ. Так, один из них (статическая модель Леонтьева) приводит к необходимости многократного решения системы линейных уравнений, сопровождающегося вычислением невязки векторов потребления и ресурсов. Каждое новое решение соответствует корректировке плана (например, вследствие применения новой технологии) и уменьшает невязку.

Принципиальная возможность решения еще не означает его практическую осуществимость. Применение обычных методов решения крупноразмерных систем линейных уравнений на ЭВМ, во-первых, требует огромного количества машинного времени (что само по себе не так уж страшно), во-вторых, что самое главное, связано с неизбежными большими интервалами времени между вычислением отдельных невязок. Именно вследствие таких больших перерывов специалисты, корректирующие план, теряют общие взаимосвязи, имеющиеся в модели, и по сути вместо целенаправленной корректировки плана получается подобие случайного поиска на целевой функции неизвестной структуры, который, как известно, дает исключительно медленную сходимость.

Иное дело, когда каждое предложение специалистов, хорошо знающих локальные зависимости в модели, вследствие применения специальных численных методов линейной алгебры проверяется немедленно (предположим, в течение 10—20 мин). В этом случае опыт и точное экономическое чутье специалистов могут привести к решению оптимизационной балансовой задачи на обозримое число итераций. Но для этого диалог между ЭВМ и специалистами нужно организовать с максимальными удобствами для последних, чтобы они имели возможность свободно, не утомляясь и действуя согласованно, провести ряд сеансов работы с ЭВМ, во время которых проиграть предложения по улучшению плана.

Диалоговый режим существенно необходим также в некоторых задачах **автоматизированного управления предприятием.** Пусть, например, на предприятии сложилась такая ситуация, что выполнить в срок поставленные перед ним задачи, следуя ранее принятому плану-графику работ, невозможно. Возникает необходимость принять оперативные решения, связанные, быть может, с интенсификацией процесса, с включением дополнительных мощностей и т. п. Любое из этих решений, как правило, приводит к затрате дополнительных средств, и заранее определить, по какому пути лучше идти в данной конкретной ситуации, в принципе невозможно, так как оба «крайние» пути, использующие противоречивые критерии максимальной экономии средств или максимальных темпов производства, имеют свои положительные и отрицательные стороны. Руководитель предприятия в таких случаях, используя свой производственный опыт и информацию, полученную из информационной подсистемы АСУП, конструирует какой-то промежуточный критерий, наиболее соответствующий, по его мнению, основной экономической и производственной задаче предприятия на данном этапе и учитывающий также социальные факторы. Свое решение он может корректировать и уточнять, получая всякий раз оперативную «обратную связь» от информационной подсистемы АСУП. Таким образом, необходимые изменения в производстве реализуются в результате своеобразного диалога человека с информационной системой.

Само собой разумеется, что эффективная реализация диалога невозможна без осуществления ряда серьезных требований к **техническим средствам и языкам.**

Применение диалогового режима в универсальных системах разделения времени не решает вопрос, так как последние не обеспечивают специалистам необходимых удобств для их работы с ЭВМ. Перспективным с этой точки зрения является создание проблемно-ориентированных миникомпьютеров и применение их на нижней ступени двух- или трехуровневых иерархических систем. Такие миникомпьютеры с соответствующими устройствами обмена обеспечивали бы наиболее естественное взаимодействие специалиста с системой. При этом традиционные методы трансляции входных программ становятся во многих случаях неприемлемыми и должны уступить место методу интерпретации входного языка. Значительный опыт в этом отношении накоплен при разработке ЭВМ серии «МИР». Уже сейчас можно четко очертить функциональные свойства входных миникомпьютеров, ориентированных на задачи проектирования, доказательство теорем, планирование и пр. Применение микропрограммной памяти на интегральных схемах предоставляет возможность изменения ориентации миникомпьютера в широких пределах.

Ясно, что организация диалога в виде обмена текстовыми директивами и сообщениями является далеко не идеальным техническим решением вопроса. «Интерфейс человека» допускает значительно более разнообразный обмен информацией со средой, чем манипуляции с клавишами пишущей машинки и чтение текста на экране. Таким образом, перед разработчиками технических средств диалога стоит задача создания различных перспективных устройств обмена. В первую очередь следует направить усилия на синтезаторы человеческой речи, во многом дополняющие визуальные устройства отображения. В свою очередь, развитие последних в настоящее время требует вывода объемных многоплановых, цветных, полутоновых и движущихся изображений. Такие задачи стоят перед разработчиками устройств вывода. Весьма перспективными, например, являются анализаторы речи и автоматы, способные читать рукописный текст в процессе его написания.

Важно отметить, что человек может применять сочетание нескольких способов обмена, не задумываясь при этом над формальным «переключением» собеседника с одного способа на другой. Нечто подобное должно предусматриваться и в диалоговых системах. В ряде случаев может оказаться целесообразным параллельное применение нескольких различных способов обмена (например, звуковое сопровождение визуальных кадров), что приведет к повышению пропускной способности и надежности канала, связывающего человека с машиной.

Серьезные проблемы возникают также при создании диалоговых языков. Основной из них является согласование противоречия между разнообразием средств, характерных для современных алгоритмических языков (часто весьма сложных), и требованиями к их максимальной лаконичности и выразительности, а также к отражению в них «человеческого образа мышления». Некоторые из этих требований нашли реализацию в языке APL/360 [5].

Представляется целесообразной многослойная структура языка, в которой на переднем плане функционируют диалоговые подмножества, ориентированные на конкретные проблемы и существенно отличающиеся от современных универсальных алгоритмических языков. В таких подмножествах, например, должен широко использоваться принцип умолчания, они должны быть «терпимы» к человеческим ошибкам, в частности к ошибкам, заключающимся в неполноте информации, содержащейся в директиве человека, к многозначности выражения мысли. Диалоговая система, сталкиваясь с подобными ситуациями, должна предпринять попытку дополнения

директивы соответствующей информацией, исходя, например, из контекста специфики решаемой задачи или даже из индивидуальных характеристик работающего на машине специалиста (такие характеристики могли бы формироваться операционной системой на основании статистики ошибок, допускаемых данным специалистом). Естественно, что во многих случаях такое доопределение невозможно и диалоговая система должна запросить соответствующую информацию (опять-таки минимальную) от человека. Сам запрос при этом должен быть построен в соответствии с правилами дидактики, включающими различного рода «подсказки», напоминания и др.

Здесь, пожалуй, имеет смысл говорить об определенной степени «равноправия» человека и машины при совместном решении задачи. Диалоговая система должна не только быть оперативной и обеспечить наглядность материала, предъявляемого пользователю, но и помочь ему уточнить условия задачи, выбрать метод ее решения, привлечь к ее решению подходящие машинные средства. Такая помощь предопределяет управление диалогом со стороны машины, а также автоматическое обучение ее пользователя в процессе решения задачи [6].

К настоящему времени можно указать лишь небольшое число систем, в которых управление диалогом осуществляется машиной. Это прежде всего обучающие системы, обеспечивающие управление пользователем, обучаемым при решении им тренировочных задач. Примером является система ПЕДАГОГ, разработанная в Институте кибернетики АН УССР для обучения языкам программирования и внедренная на машинах серии «Днепр-2». Особенности построения системы позволяют организовать обучающий диалог с пользователем, имеющим элементарные навыки работы на телетайпе и знающим вначале лишь один оператор языка директив этой системы — СЛОВАРЬ [7].

В заключение еще раз подчеркнем, что число сложных задач, требующих применения диалогового режима, непрерывно растет. От решения проблем, связанных с технической и языковой реализацией диалога, во многом зависит не только повышение эффективности использования средств вычислительной техники и быстрота решения того или иного класса задач, но в какой-то мере и общий прогресс науки, связанный с проникновением математических методов в ее естественные, технические и гуманитарные отрасли.

К этому широкое обсуждение вопросов, представленных в данной статье должно быть лишь одним из шагов на пути значительной интенсификации исследований и практических разработок в области диалоговых систем.

СПИСОК ЛИТЕРАТУРЫ

1. *Eskelson N.* Forderungen an ein Dialogsystem // Electronische Rechenanlagen.— 1971.— 5.
2. *Филиппов В. И.* Интерпретирующая система разделения времени ДИАЛОГ — БЭСМ-6 (публикуется в настоящем выпуске журнала).
3. *Борисенко Л. Г., Лаврищева Е. М.* Система диалоговой отладки ОПАЛ (публикуется в настоящем выпуске журнала).
4. *Глушков В. М., Капитонова Ю. В., Летичевский А. А.* Математическое обеспечение автоматизированной системы проектирования машин и систем (ПРОЕКТ) // Кибернетика.— 1970.— № 4.
5. *Gilman L., Rose A. J.* APL/360, An Interactive Approach // John Wiley and Sons, Inc., 1970.
6. *Press L.* Toward Balanced Man-Machine Systems // Intern. Journal of Man-Machine Studies.— 1971.— N 3.
7. *Довгялло А. М., Ющенко Е. Л.* Системы для обучения и решения задач в режиме диалога // Кибернетика.— 1973.— № 3.— С. 134—144.

ОБ АРХИТЕКТУРЕ ВЫСОКОПРОИЗВОДИТЕЛЬНЫХ ЭВМ

(Препр. Ин-та кибернетики АН УССР
№ 78—65, 1978)

Повышение производительности ЭВМ традиционной архитектуры за счет увеличения быстродействия элементов имеет свои пределы. Поэтому при построении ЭВМ высокой производительности наряду с непрерывным совершенствованием элементной базы применяются различные архитектурные средства увеличения быстродействия. Для повышения быстродействия отдельного процессора в настоящее время используются два основных метода. Первый метод, впервые реализованный в отечественных ЭВМ серии «МИР», состоит в повышении уровня внутреннего машинного языка. Увеличение быстродействия в этом случае происходит за счет двух основных причин. Во-первых, повышение уровня машинного языка приводит к сокращению длины рабочих программ и, следовательно, к уменьшению числа обращений к памяти. Во-вторых, структурная реализация сложных операторов позволяет, как правило, существенно ускорить их выполнение по сравнению с программной реализацией этих операторов на ЭВМ традиционной архитектуры с простыми внутренними языками.

Наиболее простым и достаточно гибким методом структурной реализации сложных операторов является ступенчатое микропрограммирование, также впервые реализованное на ЭВМ серии «МИР». Суть этого метода в следующем. На некотором множестве регистров с помощью соответствующих комбинационных схем реализуется некоторый набор микроопераций (т. е. преобразований на регистрах, выполняемых за один элементарный тактовый промежуток). Примерами таких микроопераций являются как традиционные микрооперации поразрядного сложения, пересылки с регистра на регистр, сдвиги и т. д., так, возможно, и новые микрооперации, не встречавшиеся ранее в инженерной практике (например, сдвиг с инвертированием четных разрядов). Такие «необычные» микрооперации возникают при применении методов формальных преобразований в автоматизированных системах проектирования ЭВМ.

С помощью заданных таким образом микроопераций строятся микропрограммы операций первого уровня, запоминаемые в нижней ступени специального микропрограммного ЗУ (МПЗУ). Во множество этих операций могут включаться обычные арифметические операции над многозначными числами и другие операции с относительно простыми микропрограммами.

Операции второго и более высоких уровней представляются все более сложными микропрограммами, запоминаемыми в МПЗУ.

Первое отличие таких сложных операторов от программно-реализуемых макрооператоров заключается в том, что переходы с одного уровня управления на другой осуществляются без обращения в ОЗУ за счет использования регистров, встроенных в устройство управления. Второе отличие определяется возможностью использования микропрограммами любого уровня не

только машинных команд (как это происходит при обычной программной реализации макрооператоров), но и любой микрооперации из исходного (реализованного в арифметико-логическом устройстве АЛУ) набора.

Ясно, что указанные отличия способствуют увеличению быстродействия процессора при выполнении соответствующих макрооператоров. Например, пусть при выполнении макрооператора $y := \cos x$ значение косинуса вычисляется путем его разложения в ряд $1 - x^2/2 + \ldots$ Если сдвиг на одном из регистров АЛУ не включен в число машинных команд, то при обычной реализации в виде стандартной подпрограммы деление $x^2/2$ должно быть выполнено по полному циклу (с запоминанием константы 2 в ОЗУ). При ступенчатой микропрограммной реализации в этом случае можно использовать лишь одну микрооперацию сдвига.

В первых ЭВМ, использовавших структурную реализацию сложных алгоритмических языков, микропрограммные ЗУ выполнялись обычно в виде пассивного (одностороннего) запоминающего устройства (ПЗУ) с неизменным набором микропрограмм. Причина такого решения заключалась в том, что в то время ПЗУ были быстрее и, самое главное, существенно дешевле, чем ОЗУ. Сейчас в связи с появлением полупроводниковых оперативных ЗУ на БИС становится целесообразным для запоминания микропрограмм использовать не ПЗУ, а ОЗУ. Вследствие этого оказывается возможным (путем перезагрузки микропрограмм) быстро перестраивать язык (набор операций и макроопераций) процессора. Подобные процессоры получили название процессоров с гибкой архитектурой, хотя, разумеется, гибкость при этом имеет не весь процессор, а лишь некоторая его часть.

Второй метод повышения быстродействия отдельных процессоров был разработан в 60-е годы в США. Он получил название конвейерного, или магистрально-трубопроводного метода. Его реализация предусматривает усложнение структуры АЛУ. При этом АЛУ представляется в виде цепочки (конвейера) устройств, специализирующихся на выполнении отдельных составных частей машинных операций. Например, после выборки операндов для очередной команды некоторым блоком конвейера исполнение команды передается на следующие блоки, а данный блок используется для выборки операндов для следующей команды. Аналогично исполняется сама команда: например, сложение чисел с плавающей запятой может быть разложено на более мелкие операции (выравнивание порядков, сложение мантисс, нормализация результата), выполнение которых поручается различным блокам конвейера.

В предельном случае конвейер (когда каждый блок выполняет отдельную операцию) при благоприятных условиях (независимости операндов в последовательных командах) может выдавать результаты вычисления с частотой, равной времени выполнения одной микрооперации, а не машинной команды, как в обычном процессоре.

Описанные методы повышения быстродействия отдельного процессора являются универсальными в том смысле, что их эффективность относительно мало зависит от вида решаемых задач (при предположении, что структурно-реализуемый алгоритмический язык в первом случае и блоки конвейера во втором достаточно универсальны).

За счет более узкой специализации машинных языков и процессорных элементов можно строить специализированные процессоры, обладающие особо высокой производительностью на том или ином классе задач. Одним из наиболее известных типов специализированных процессоров являются векторно-матричные процессоры, позволяющие выполнять операции над отдельными элементами векторов и матриц одновременно. Служащие для

выполнения таких операций процессорные элементы являются по сути простейшими процессорами. Поэтому векторно-матричные процессоры можно рассматривать и как специализированные многопроцессорные ЭВМ.

Повышение производительности процессоров довольно остро ставит вопрос о скорости обмена процессоров с ОЗУ. Существует ряд способов увеличивать эту скорость. Одним из простейших является способ увеличения количества разрядов в ячейках ОЗУ, позволяющих за одно обращение к ОЗУ пересылать несколько машинных слов. Хотя время цикла в ОЗУ при увеличении разрядности растет, однако не очень быстро, что с лихвой перекрывается краткостью выборки.

Второй способ — это так называемое перекрытие (overlapping). При этом ОЗУ составляется из нескольких блоков, к которым возможно независимое (одновременное) обращение. Поскольку в большинстве случаев (при пересылке как команд, так и данных) приходится обращаться к последовательно расположенным друг за другом ячейкам памяти, то это расположение приспосабливается к возможности независимого обращения к последовательным ячейкам ОЗУ. Так, если 1-я ячейка ОЗУ совпадает с 1-й ячейкой 1-го блока ОЗУ, то 2-й ячейкой считается не 2-я ячейка 1-го блока (рис. 1), а 1-я ячейка 2-го блока. При n блоках n-я ячейка совпадает с 1-й ячейкой n-го блока, а $(n + 1)$-я ячейка — со 2-й ячейкой 1-го блока.

Подобная нумерация ячеек памяти позволяет организовать одновременную выборку из n ячеек ОЗУ и тем самым привести в соответствие скорость обмена с ОЗУ скорости обработки информации в процессоре.

Параллельная выборка из ОЗУ предполагает наличие в процессоре специальной быстродействующей буферной памяти, обеспечивающей обмен информацией как с процессором, так и с ОЗУ. Минимальный размер буфера равен размеру одновременной выборки из ОЗУ. Однако с учетом частого использования результатов предшествующих операций машинной программы в последующих оказывается целесообразным увеличивать размер буфера до тех пор, пока это не начнет приводить к существенному уменьшению его быстродействия. Такой буфер принято называть сверхоперативной памятью (СОЗУ). Таким образом, в ЭВМ большой производительности используется фактически 4-уровневая память: регистры процессора, СОЗУ, ОЗУ и внешняя память на магнитных носителях. Иногда к ним добавляется 5-й уровень массовой памяти (mass memory), промежуточный по скорости обмена между ОЗУ и внешней памятью.

Многие задачи невычислительного характера, связанные с выборкой и переупорядочением больших массивов информации, требуют особой организации памяти, когда обращение к тем или иным ячейкам ОЗУ происходит не по адресам ячеек, а по признакам, закодированным определенными разрядами хранящихся в них слов. Память с возможностью такого обращения называется ассоциативной. Одна из возможных форм организации подобной памяти — матричное ОЗУ с выборкой как по строкам, так и по столбцам. При обращении к строке выбирается отдельное (обычно достаточно длинное)

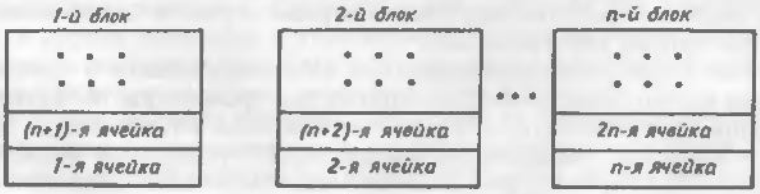


Рис. 1.

слово, а при обращении к столбцу — один и тот же разряд всех слов, запомненных в матрице (рис. 2).

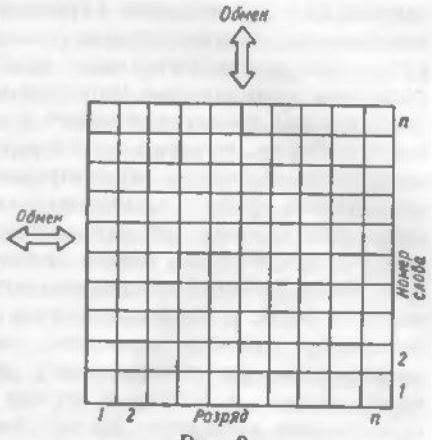


Рис. 2.

На базе ассоциативного ЗУ создаются специализированные ЭВМ, обладающие сверхвысоким быстродействием на задачах, связанных с сортировкой, подборкой, выборкой по признакам и другими операциями на больших информационных массивах. Для других применений оказываются эффективными иные способы организации памяти, например стековый.

При большой скорости обмена с ОЗУ к нему можно подключать несколько процессоров (снабженных обычно собственными небольшими сверхоперативными ЗУ). Тем самым возникают универсальные мультипроцессорные системы с общей памятью. Разумеется, при этом число процессоров обычно бывает небольшим, так что серьезного распараллеливания вычислительного процесса осуществить подобным путем не удается. Значительно большие возможности распараллеливания возникают в том случае, когда в систему объединяются процессоры, снабженные собственными оперативными ЗУ. Теоретически в подобные системы можно объединить какое угодно число процессоров. На практике, однако, возникают трудности, связанные с эффективной коммутацией процессоров и управлением системой. Эти трудности преодолеваются относительно легко на специальных классах операций, например векторно-матричных. В общем же случае принятые до сих пор методы построения мультипроцессорных систем дают относительно медленный (логарифмический) рост суммарной производительности системы в зависимости от составляющих ее процессоров. Это делало невыгодным объединение в систему большого числа процессоров.

В то же время успехи микроэлектроники и особенно появление однокристаллных микропроцессоров делают задачу построения больших мультипроцессорных систем весьма актуальной. Чтобы справиться с указанной выше трудностью и получить высокоэффективные мультипроцессорные системы, состоящие из сотен и даже тысяч микропроцессоров, предлагается существенно изменить принципы коммутации и управления в таких системах.

При создании сложных мультипроцессорных систем на базе микропроцессоров целесообразно руководствоваться принципом автономно-иерархической блочности. Суть его в том, что система составляется из блоков разных уровней иерархии, каждый из которых может иметь свое собственное независимое применение в автономном режиме. Блоки, о которых идет речь, определяются сложившимися конструктивными ячейками: кристалл, плата, сборка, шкаф, система. Содержание, вкладываемое в каждый блок, определяется достигнутым уровнем интеграции. При нынешнем уровне интеграции на одной плате целесообразно реализовать полноразрядный микрокомпьютер с гибкой архитектурой и собственным небольшим ОЗУ. На этой же плате должен быть организован выход в стандартный (скажем, 9-битовый) канал для дуплексной связи с периферийными устройствами и другими микрокомпьютерами, помещаемыми в одну сборку. Желательно на этой же плате иметь внешнее ЗУ на магнитных доменах или приборах с зарядовой

связью для организации автономной виртуальной памяти каждого микрокомпьютера. Такие одноплатные микрокомпьютеры могут использоваться автономно в измерительных приборах простейших АСУ ТП. Несколько подобных компьютеров (которые мы будем называть рабочими) с помощью специальных коммутационного и управляющего процессоров объединяются в одну сборку, которая может (при оснащении ее специальной периферией) выполнять функции многопроцессорной миниЭВМ. При этом сборкам желательно придать самостоятельное конструктивное оформление, позволяющее использовать их как автономно, так и встраиваемыми в следующий иерархический блок, каким является шкаф.

Число рабочих микрокомпьютеров в сборке относительно небольшое — порядка 5—8, в крайнем случае — порядка 10—16. Коммуникационный процессор должен позволять организовывать одновременную передачу информации по стандартному каналу между любыми парами микрокомпьютеров в сборке и, кроме того, обеспечивать несколькими дополнительными каналами выход на следующий (внутришкафный) уровень коммутации.

При восьми коммутируемых микрокомпьютерах для организации дуплексной связи коммутационный процессор должен иметь $18 \times 8 = 144$ входа, не считая входов для дополнительных каналов и энергопитания. Поэтому его нельзя монтировать на плате обычного типа, поскольку здесь будут превышаться возможности используемых ныне разъемов. В связи со сказанным представляется целесообразным размещать коммуникационный процессор в операционной жесткой плате с вмонтированными в нее разъемами для плат рабочих микрокомпьютеров, помещаемых перпендикулярно к этой плате. Подобная плата, которую мы будем называть коммуникационной, крепится на конструкции сборки болтами (что позволяет в случае необходимости снять и заменить ее) и снабжается дополнительными разъемами (обращенными вовне сборки) для гибкого кабеля, через которые осуществляется связь с другими сборками и подача питания.

Наряду с рабочими микрокомпьютерами в сборку помещается также управляющий процессор (с собственными ОЗУ), организующий управление вычислительным процессом внутри сборки и взаимодействие через коммуникационный процессор с управляющими процессорами более высоких уровней иерархии. Управляющий процессор может монтироваться на коммуникационной плате или на отдельной рабочей плате взамен одного из рабочих микрокомпьютеров. Внешнее конструктивное оформление сборки желательно иметь таким, чтобы подобным образом могли бы быть оформлены внешние ЗУ миниЭВМ (минидиски, кассетные магнитофоны), а также устройства ввода — вывода на гибких дисках. Подобное конструктивное однообразие позволит гибко компоновать шкафы, в частности снабжать каждую сборку миниЭВМ своей собственной внешней памятью, что значительно увеличивает возможности распараллеливания вычислительных процессов и, что особенно важно, процессов управления ими.

Каждый шкаф должен снабжаться коммуникационным и управляющим процессорами второго уровня иерархии. Они могут монтироваться как в отдельной сборке, так и в виде специальной съемной жесткой платы с соответствующим числом кабельных разъемов. Снабженный соответствующей периферией, каждый такой шкаф может использоваться как средняя мультипроцессорная ЭВМ. Для получения ЭВМ большой производительности и суперЭВМ нужно объединять (в одном или нескольких уровнях иерархии) соответствующее число шкафов. Этот же принцип может быть применен при объединении удаленных ЭВМ в сети.

В описанной конструкции просматривается первая основная идея, позволяющая строить высокоэффективные мультипроцессорные системы. Она состоит в том, что в отличие от общих шин для быстрой передачи информации, применяющихся в традиционных мультипроцессорных системах, строится более медленная, но гораздо более гибкая коммуникационная сеть, напоминающая обычную телефонную сеть связи.

Сама по себе эта идея еще не обеспечивает эффективной работы мультипроцессорной системы, так как низкая скорость обмена при обычных методах распараллеливания обработки информации может свести «на нет» преимущества параллельной работы большого числа процессоров. Поэтому ее необходимо дополнить такой идеей организации параллельной обработки, которая уменьшила бы требования к скорости обмена информацией в системе.

Суть второй идеи состоит в том, чтобы в качестве основного принципа параллельной обработки информации был избран принцип макроконвейера. В отличие от описанного выше микроконвейера, распараллеливающего выполнение отдельных микроопераций, макроконвейер настраивается на выполнение на каждом его рабочем месте достаточно крупных макрооператоров, способных составить серьезную загрузку для отдельных микрокомпьютеров мультипроцессорной системы.

Для настройки макроконвейера необходимо выделить соответствующее количество микрокомпьютеров, организовать необходимые связи между ними и заслать в быструю микропрограммную память каждого микрокомпьютера микропрограммы, наилучшим образом приспосабливающие его для предполагаемой работы. Все это требует довольно большой работы по предварительному планированию и последующему управлению вычислительным процессом и нуждается в соответствующих новых идеях по организации такого планирования и управления, описываемых далее.

Здесь мы лишь отметим, что при исключении лишь некоторых случаев управления в реальном масштабе времени линейные участки программ не требуют распараллеливания. Причину этого понять нетрудно. До сих пор самые длинные программы для ЭВМ не превышали 1—2 млн машинных команд. При современном быстродействии обычная однопроцессорная ЭВМ выполнит линейную программу такого размера за доли секунды. Ясно, что в дальнейшем рост сложности программ не может идти слишком быстрыми темпами, поскольку он в конечном счете ограничивается возможностями человека. Поэтому основное внимание при распараллеливании вычислений следует уделять циклам, особенно вложенным друг в друга. Для этих последних сегодня разработаны методы распараллеливания, не опирающиеся на содержательный смысл задачи (как, например, это происходит при принятых сейчас методах распараллеливания векторно-матричных операций). С некоторыми из таких методов можно ознакомиться, скажем, по работе Лампорта. Правда, техника, изложенная Лампортом, несколько удалена от практических задач распараллеливания, поскольку она не учитывает двух важных обстоятельств: ограниченности числа процессоров и необходимости достаточно полной загрузки каждого из них. При некотором усовершенствовании методов Лампорта этот недостаток можно устранить, получив методы, пригодные для практического распараллеливания. При таком усовершенствовании во многих важных для практики случаях (например, при решении краевых задач математической физики и др.) распараллеливание производится так, что количество информационных обменов между составляющими макроконвейер микрокомпьютерами существенно меньше общего числа выполняемых ими операций. А это обстоятельство как раз и является

решающим для получения высокой производительности конвейера при относительно медленной связи между составляющими его частями.

В случаях, когда подобное благоприятное условие не имеет места (например, для обычных методов решения задачи Коши обыкновенных дифференциальных уравнений), можно предложить новые методы решения задач, специально приспособленные для организации высокоэффективных макроконвейеров.

Третья, основная, идея состоит в распараллеливании процессов планирования и управления в многопроцессорных системах. В известных ныне многопроцессорных системах управление централизуется в одном процессоре, в результате чего организация эффективной работы большого числа взаимосвязанных рабочих процессоров (как это требуется при создании макроконвейеров) очень замедляется и затрудняется. Кроме того, в вычислительной технике в целом исторически так сложилось, что при управлении вычислительным процессом операционная система имеет дело лишь с рабочей программой. Все же остальные материалы, получаемые (и документирующиеся) в процессе проектирования и изготовления программы (технические задания, блок-схемы, описания в алгоритмических языках и т. п.), при этом полностью забываются.

Предлагаемая идея распараллеливания управления в мультипроцессорной системе состоит в использовании в управляющих процессорах разных иерархических уровней всей полезной для управления документации, получаемой в процессе проектирования рабочих программ. Реализация этой идеи требует в значительной мере перестроить весь процесс проектирования и отладки программ, а также произвести соответствующие изменения документации с целью ее эффективного использования для улучшения управления в мультипроцессорных системах.

Распараллеливанию управления способствует также иерархическое управление ресурсами в описанной выше конструкции мультипроцессорной системы. В управляющих процессорах высших уровней может не происходить точный учет занимаемых и освобождаемых ресурсов в более низких уровнях, подведомственных младшим управляющим процессорам. Это свойство проявляется с особой силой при включении в состав плат и сборок своих периферийных ЗУ.

Еще отметим, что описанная конструкция включает в себя только основную (базовую) аппаратуру высокопроизводительных мультипроцессорных систем. Она может дополняться различного рода спецпроцессорами (микроконвейерами, матричными и др.), обобществленными ОЗУ и другими устройствами.

Для эффективной работы с периферийными устройствами, особенно с обобществленными внешними ЗУ, в ряде случаев потребуется менять формы организации файлов. Например, для параллельной работы рабочих процессоров с большим файлом может оказаться целесообразным размещать разные его куски на различных внешних ЗУ. Есть еще ряд вопросов, требующих новых по сравнению с ЭВМ традиционной архитектуры подходов и решений.

ОСНОВНЫЕ АРХИТЕКТУРНЫЕ ПРИНЦИПЫ ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ ЭВМ

(Проблемы вычислительной техники: спецвыпуск МЦНТИ.— М. : 1981)

Реализация архитектурных принципов построения ЭВМ, известных как принципы фон Неймана, открыла эру вычислительной техники.

Первый принцип — максимально возможная простота процессора и связанная с ней простота машинного языка системы команд ЭВМ.

Второй принцип — последовательное (командно-адресное) управление вычислительным процессом, при котором команды извлекаются из оперативного запоминающего устройства (ОЗУ) и исполняются (в процессоре) последовательно одна за другой. Данные (операнды) для каждой команды хранятся в том же ОЗУ по адресам, указываемым в команде. Взаимосвязь (интерфейс) процессора и ОЗУ ограничивает возможности этих устройств, являясь своеобразным «узким горлышком» всей архитектуры фон Неймана.

Третий принцип — адресная организация ОЗУ с последовательной (линейной) структурой адресов и фиксированным размером ячеек. Такая организация ОЗУ плохо согласуется со сложными структурами данных, которые характерны для большинства задач, решаемых на современных ЭВМ.

Четвертый принцип — использование процессора для управления всеми вспомогательными операциями, обеспечивающими вычислительный процесс: обменами между ОЗУ и внешней памятью, а также вводом — выводом информации.

Пятый принцип — жесткость архитектуры, исключающей любое изменение конфигурации совокупности устройств, входящих в состав ЭВМ, как с точки зрения изменения состава устройств, так и с точки зрения изменения взаимосвязей между ними.

Сегодня недостатки архитектуры, строящейся на базе принципов фон Неймана, очевидны. Однако не следует забывать, что они в свое время сыграли весьма прогрессивную роль и при существующем тогда уровне техники были единственно возможными. Ввиду громоздкости, дороговизны и малой надежности логических элементов, строившихся на электронных лампах, конструкторы ЭВМ вынуждены были максимально ограничивать число таких элементов. Структура ОЗУ также лимитировалась имевшимися реальными техническими возможностями.

Однако уже на ранних ступенях развития электронной вычислительной техники было ясно, что прогресс электронной технологии рано или поздно приведет к возможности разработки принципиально новых архитектурных решений. Если предположить, что конструктор может объединять в систему не несколько тысяч логических элементов, как это было в эпоху электронно-ламповой технологии, а многие десятки миллионов (причем на число соединений этих элементов практически не накладывается никаких ограничений), то лучшими архитектурными решениями для ЭВМ будут мозгоподобные структуры. Характерной особенностью их является слияние памяти с обра-

боткой данных: данные обрабатываются одновременно по всей памяти с максимально возможной степенью распараллеливания всех операций. Перспективность подобных структур при соответствующем прогрессе микроэлектроники отмечалась автором на конференции в Киеве уже в 1959 г. Подчеркнем, что речь идет именно о мозгоподобных структурах, а не о точном копировании мозга, в котором эффективно распараллеливаются далеко не все операции (в частности, в мозге плохо распараллеливаются собственно вычислительные операции).

Хотя мозгоподобные структуры с параллельными процессами, управляемыми многими потоками данных и команд, несомненно, представляют собой высший уровень развития архитектур ЭВМ, однако на нынешнем этапе электронной технологии полная и бескомпромиссная их реализация является пока преждевременной. Необходимы компромиссные решения, представляющие собой переходные этапы к мозгоподобным структурам будущего на основе разумного отступления от принципов фон Неймана.

Подчеркнем, что повышение производительности ЭВМ в нашем понимании отнюдь не сводится к простому увеличению их быстродействия. Речь идет о производительности относительно полного цикла решения задачи на ЭВМ, включая программирование, а также подготовку и ввод исходных данных. С этой точки зрения любую ЭВМ можно рассматривать как своеобразную фабрику (или в простейших случаях мастерскую) по переработке информации. Эта аналогия выходит за рамки простых популяризаторских целей и является в действительности настолько глубокой, что может служить (и уже не раз послужила) источником новых идей в развитии архитектуры ЭВМ и вычислительных комплексов.

Мы используем эту аналогию для того, чтобы проследить генезис основных идей совершенствования архитектуры ЭВМ с целью повышения их производительности. Исходная неймановская архитектура при этом уподобляется примитивной кустарной мастерской, где кустарь-одиночка (процессор) сам выполняет все работы: достает исходные материалы (вводит данные) и инструкции по их обработке (программу), располагает эти материалы и инструкции по полочкам склада (ячейкам ОЗУ), выбирает в установленном порядке инструкции и материалы, подвергая их обработке в соответствии с этими инструкциями, раскладывает полученные полуфабрикаты по полочкам склада и выдает готовые изделия. Если емкости основного (оперативного) склада (ОЗУ) недостаточны, процессор перемещает менее нужные в настоящий момент материалы и инструкции на дополнительные, более емкие, но более труднодоступные склады внешних запоминающих устройств (ВЗУ), освобождая место в основном складе.

Подчеркнем, что в соответствии с первым принципом фон Неймана сам кустарь (процессор) весьма далек от совершенства, он владеет навыками лишь простых операций и не способен запоминать и исполнять несколько таких операций подряд: правила выполнения каждой из них он должен каждый раз извлечь со склада, прочесть и лишь после этого исполнить. Поэтому даже в случае очень быстрого темпа проведения отдельных операций и обращений к складу производительность подобной кустарной мастерской при выполнении сложных заданий может быть сравнительно небольшой.

Переходя к рассмотрению принципов повышения производительности ЭВМ, мы для простоты будем останавливаться лишь на основных (базовых) из них, подразумевая как нечто само собой разумеющееся, что в реальных архитектурных решениях могут использоваться самые разные комбинации этих принципов.

Первый базовый принцип повышения производительности ЭВМ — рост «интеллектуальности» процессора, т. е. придание ему способности выполнять по одной команде достаточно сложные операции. Повышая тем самым уровень машинного языка, мы сразу решаем три важные задачи. Во-первых, резко упрощаем программирование, не используя при этом дополнительных затрат машинного времени на трансляцию программ с языков высокого уровня на примитивные (неймановские) машинные языки. Во-вторых, уменьшая длину программы, мы сокращаем время на обращение процессора к ОЗУ и, экономя память, сокращаем непроизводительные пересылки между ОЗУ и ВЗУ. В-третьих, специально конструируя процессор для выполнения более сложных операций, можем принять дополнительные меры для увеличения его быстродействия.

Идея повышения «интеллектуальности» процессора была впервые не только высказана (1959 г.), но и реализована в СССР в ЭВМ «МИР-1» (серийный выпуск с 1965 г.). В 1966 г. Институтом кибернетики АН УССР разработан технический проект большой ЭВМ «Украина», в которой были заложены возможности схемной интерпретации многих появившихся к тому времени проблемно-ориентированных алгоритмических языков.

Примерно в это же время (1967—1968 гг.) идея повышения уровня «интеллектуальности» процессоров была воспринята американской фирмой «Burroughs», а в 70-е годы она получила уже всеобщее признание и широкое распространение.

Отметим, что на первом этапе реализации усложнялось в основном лишь устройство управления (УУ) процессора за счет включения в его состав быстродействующего иерархического запоминающего устройства (ЗУ) для запоминания микропрограмм достаточно сложных операций. Уже в машине «МИР-1» в УУ включались микропрограммы столь сложных операций, как вычисление определенных интегралов, факториала, не говоря уже о вычислениях простейших элементарных функций. По сравнению с обычной (программной) реализацией таких операций производительность ЭВМ значительно возросла.

В гораздо большей степени этот эффект проявился в ЭВМ «МИР-2», где в УУ были встроены ступенчатые микропрограммные ЗУ, реализующие в качестве элементарных операций сложные аналитические преобразования. Выигрыш в производительности был так велик, что для многих конкретных аналитических задач время их решения на ЭВМ «МИР-2» оказывалось (с учетом времени, необходимого для трансляции) сравнимым с временем их решения на машинах неймановской архитектуры с номинальным быстродействием (на элементарных операциях), в сотни раз превосходившим быстродействие машины «МИР-2».

Второй этап реализации идеи повышения способностей процессоров связан с усложнением арифметических или (как их начали к этому времени называть) арифметико-логических устройств (АЛУ). Это позволило реализовать дополнительные возможности по распараллеливанию и убыстрению выполнения сложных операций. Кроме того, развитие микроэлектроники в 70-е годы привело к возможности замены пассивных (долговременных) микропрограммных ЗУ оперативными. Это вызвало к жизни так называемую мягкую архитектуру процессоров, т. е. набор выполняемых ими операций может оперативно меняться не только при переходе от одних классов задач к другим, но даже в процессе решения одной и той же задачи. Тем самым были задействованы дополнительные резервы увеличения производительности процессоров.

Второй базовый принцип повышения производительности ЭВМ — освобождение процессора от выполнения таких вспомогательных операций, как управление вводом/выводом данных, а также обменами между ОЗУ и ВЗУ. Для этого в состав ЭВМ наряду с неймановским центральным процессором вводятся специализированные процессоры, называемые обычно каналами, или периферийными процессорами. Кроме того, определенная часть функций управления периферийным оборудованием (устройствами ввода — вывода и внешней памяти) передается специальным контроллерам, сопрягаемым с этим оборудованием.

С учетом «производственной аналогии» это означает появление в «мастерской» по переработке информации наряду с основным «мастером» (центральным процессором) «вспомогательного рабочего персонала». Освобождая время центрального процессора для основных производственных операций, подобное усовершенствование вместе с тем налагает на него дополнительную функцию координации работы всех остальных функциональных звеньев. Сюда входит не только выдача заданий специализированным процессорам, но и разрешение конфликтов при их обращении к общим ресурсам, какими являются ресурсы оперативной памяти и самого центрального процессора. Для осуществления подобной координации строится сложная система управляющих программ — так называемая операционная система (ОС).

Принцип децентрализации управления периферийным оборудованием уже в 50-е годы в той или иной степени был реализован многими фирмами, так что в настоящее время не так просто установить первоначальное авторство этой идеи. Отметим, что в полной мере для повышения производительности ЭВМ этот принцип может быть использован лишь в комбинации с другим принципом, получившим название мультипрограммирования.

Смысл данного принципа в том, что в ЭВМ вводятся одновременно несколько заданий. В результате при хорошем планировании работы (что является задачей ОС) удается более полно загрузить оборудование и тем самым повысить суммарную производительность ЭВМ: пока центральный процессор занят решением одной задачи (для которой команды и данные находятся в ОЗУ), периферийные процессоры занимаются подготовкой (вводом и передачей в ОЗУ) данных и программ для других задач, а также выводом готовых результатов. Задачи ОС при этом усложняются за счет необходимости управлять переключениями и распределениями ресурсов между различными процессами (задачами), выполняемыми ЭВМ.

Реализация принципа децентрализации управления периферийным оборудованием и обменом информацией внутри ЭВМ потребовала введения принципа модульности архитектуры ЭВМ, что позволило в какой-то мере отступить от абсолютной жесткости архитектуры. Принцип модульности коснулся прежде всего ОЗУ, которое стало набираться из отдельных блоков: каждый из них допускает обращения, не зависимые от других блоков. Это позволило уменьшить конфликты обращений к ОЗУ и связанные с этим потери производительности. Аналогично (по модульному принципу) во многих ЭВМ организуются каналы связи с периферийным оборудованием, которое, в свою очередь, объединяется в отдельные модули.

Наличие многих независимых устройств ввода — вывода позволяет резко увеличить производительность вычислительной системы на этапах обмена информацией. Для многочисленных задач с относительно небольшим числом (вычислительных) операций по отношению к числу операций ввода — вывода (к ним относится большинство задач управления реальны-

ми объектами, задач обработки экспериментальных данных и др.) это обстоятельство может иметь решающее значение для увеличения производительности всей системы. Однако на практике встречается немало важных задач иного рода, для которых решающее значение имеет производительность ЭВМ (вычислительной системы) по основным операциям.

Известны два основных пути решения этой проблемы: повышение производительности центрального процессора в однопроцессорных вычислительных системах и переход к мультипроцессорным системам.

Основной принцип для первого пути сводится к расчленению функций центрального процессора и возложению их выполнения на специализированные процессорные элементы, объединенные в те или иные технологические линии и участки. Простейшей линией такого рода является конвейер для реализации обычных неймановских команд. Таким командам свойственна одна и та же, не зависящая от типа команды последовательность операций по их исполнению. Например, для трехадресных команд эта последовательность имеет следующий вид: выборка команды из ОЗУ, выделение из нее адресов двух входных операндов, выборка этих операндов из ОЗУ, выполнение операций над ними с одновременным выделением из кода команды адреса, по которому должен быть заслан результат операции, и, наконец, фактическая его засылка в ОЗУ по этому адресу. Для выполнения этой последовательности операций создается линия типа «конвейер».

При работе такого конвейера используется так называемый принцип синхронной накачки. Поскольку результат выполнения очередной команды, как правило (за исключением специальных команд перехода), не влияет на адрес выбираемой за ней следующей команды, то эта выборка выполняется первым процессорным элементом конвейера сразу же после передачи текущей команды второму процессорному элементу. При передвижении же ее к третьему элементу первый осуществляет выборку следующей команды и т. д.

При этом достигается существенное увеличение производительности по сравнению с обычным (неймановским) процессором: при равномерном распределении нагрузки между процессорными элементами — во столько раз, сколько элементов в конвейере. Правда, это будет лишь в том случае, когда выполняемая программа линейна, т. е. не содержит условных или безусловных переходов. При каждом таком переходе приходится производить перенакачку конвейера, что, разумеется, приводит к потере быстродействия. Эти потери тем больше, чем больше команд перехода в исполняемой программе и чем больше длина конвейера.

Для конвейеров небольшой длины (порядка 10 элементов), используемых в современных ЭВМ для большинства встречающихся на практике задач, эти потери относительно невелики.

Конвейерный метод восходит к английской ЭВМ «Атлас». Он получил применение и дальнейшее развитие практически во всех сверхбыстродействующих ЭВМ (IBM-360/195, STAR-100, ASC, «Эльбрус» и др.).

Дальнейшим развитием простого конвейерного метода является магистральная обработка. В дополнение к простому конвейеру для неймановских команд она позволяет организовывать конвейеры и системы взаимосвязанных конвейеров не только для операций обмена процессора с ОЗУ, но и для выполнения операций внутри самого процессора. Однако в этом случае приходится считаться с тем, что конвейеры для выполнения различных операций (например, операций сложения и умножения), хотя и содержат общие элементы, в целом различны.

Если строить отдельный конвейер для выполнения каждой операции, то получаются крайне неэкономичные конструкции с малыми коэффициентами использования оборудования. Выход из этого состоит в том, чтобы, оперативно перестраивая связи между отдельными элементами конвейеров, быстрее приспосабливать конвейерные линии к меняющейся ситуации. Один из способов такой перестройки — принцип карусели — впервые был применен во французской ЭВМ «Гамма-60» в начале 60-х годов и развит затем в работах M. J. Flynn в 1966—1972 гг.

Принцип гибкой архитектуры системы конвейеров, улучшая коэффициент использования оборудования, вместе с тем приводит к определенным потерям времени на перестройку, которые, к сожалению, далеко не всегда можно компенсировать упреждающим планированием. Поэтому в реальных магистральных структурах обычно идут на некоторый компромисс между степенью гибкости перестройки архитектуры процессора и ухудшением использования оборудования.

Отметим, что по мере увеличения «интеллектуальности» процессоров и, следовательно, усложнения выполняемых ими операций возможности увеличения их производительности на основе магистрального принципа обработки существенно возрастают. И дело здесь не просто в увеличении длины конвейеров, а в появлении новых возможностей распараллеливания вычислительного процесса на уровне машинных операций. Нетрудно понять, например, как можно ускорить выполнение операций вычисления факториала или определенного интеграла на системах параллельных конвейеров с объединяющим их заключительным магистральным блоком. То же относится к большинству формульных вычислений и к другим макрооперациям современных «интеллектуальных» процессоров.

Отметим также, что принцип магистральной обработки применим не только в центральных процессорах. Во многих системах автоматизации технологических процессов, обработки экспериментальных данных и испытаний в реальном масштабе времени широкое применение для целей первичной обработки находят магистральные предпроцессоры, которые соединяют систему датчиков (измерительных приборов) с ЭВМ, осуществляющими вторичную обработку получаемых данных. Выработаны международные стандарты конструктивного оформления подобных устройств (системы «Камак», приборный интерфейс и др.). Их роль заключается в том, чтобы максимально упростить и ускорить подготовку исходных данных для ЭВМ и тем самым увеличить производительность всей системы сбора и обработки данных.

Дальнейшее развитие принципа магистральной обработки приводит к включению в магистрали специализированных (функционально-ориентированных) АЛУ и специализированных процессоров. Такие АЛУ появились еще вне связи с магистральным принципом обработки. Примерами могут служить умножитель ЭВМ «Стрела» или блоки арифметики с плавающей запятой в ЭВМ систем IBM-360 и IBM-370.

Принципиально новой является магистральная организация самих функционально-ориентированных АЛУ и процессоров. Примером могут быть специализированные векторные и матричные процессоры. Первым полупромышленным матричным процессором высокой производительности был процессор «Иллиак-IV» Иллинойсского университета (1972 г.).

Процессорные элементы матричного процессора способны выполнять лишь простейшие арифметические операции над числами (скалярами). Однако, будучи соединенными в матричную структуру, они оказываются способными осуществлять быстрые пересылки информации между соседни-

ми (по вертикалям и горизонталям) элементами. Организуя согласованное (синхронное) управление вычислениями и пересылками, на такой структуре можно реализовать высокораспараллельные вычислительные процедуры определенных классов.

Например, на матричной структуре осуществляется эффективное распараллеливание сеточного метода решения плоской краевой задачи для уравнения Лапласа. Как известно, рекуррентная формула для этого метода имеет вид $x_{i,j}^{(k+1)} = 1/4\,(x_{i-1,j}^{(k)} + x_{i+1,j}^{(k)} + x_{i,j-1}^{(k)} + x_{i,j+1}^{(k)})$. Следовательно, на каждом очередном шаге итерации она предусматривает пересылку очередных значений скаляров, каждый элемент матрицы на четырех соседних с ним элементов и нахождение среднего арифметического этих скаляров.

Поскольку структура связей между данными в рассмотренной задаче (выражаемая приведенной формулой) соответствует структуре связей процессорных элементов в матричном процессоре, эта задача эффективно использует все процессорные элементы. При ее решении ускоряется (по сравнению с однопроцессорной схемой решения) в n раз, где n — число процессоров в матрице.

Однако при иной структуре связей между данными (например, в случае трехмерной краевой задачи для уравнения Лапласа) соответствие структур нарушается. В результате, во-первых, резко усложняется программирование задач, во-вторых, значительно ухудшается использование оборудования, поскольку большинство процессорных элементов будет работать не на вычисление, а на связь. Вследствие этого производительность матричного процессора резко уменьшается. При обычных (скалярных) же вычислениях его производительность (в однопрограммном режиме) падает до производительности одного процессорного элемента (в случае ЭВМ «Иллиак-IV» — на 2 порядка).

Вводя дополнительные связи между элементами матриц (как это делается, например, в ЭВМ типов ПС-2000 и ПС-3000), можно расширить класс задач, допускающих эффективное распараллеливание. Однако машина при этом останется специализированной в том смысле, что для большинства задач ее производительность будет падать до производительности одного процессорного элемента.

В целом за счет объединения достаточно большого числа процессорных элементов в специализированные магистральные структуры в настоящее время не составляет большого труда увеличить производительность таких структур для соответствующих (как правило, достаточно узких) классов задач в сотни раз (по сравнению с однопроцессорным вариантом).

Кроме векторно-матричных процессоров достаточно большое распространение нашли, например, фурье-процессоры и ряд других специализированных процессоров высокой производительности.

Подчеркнем, однако, что само построение специализированной магистральной структуры представляет собой лишь начальную (причем самую простую) часть общей задачи повышения производительности системы для большинства задач, для которых она предназначена и в принципе хорошо приспособлена. Весь процесс упрощается, если структура данных в процессе решения задачи остается неизменной (изменяются лишь их численные значения), все данные размещаются в распределенной памяти процессорных элементов системы, а исходные данные уже введены в нее. Если же хотя бы одно из этих условий не соблюдено, возникают трудности, которые можно устранить лишь специальными формами организации интерфейсов с ОЗУ и ВЗУ, представления данных в иерархической памяти, упреждающего

планирования указанных форм и соответственно адекватных программных средств.

Чтобы понять природу возникающих здесь трудностей, достаточно рассмотреть задачу нахождения скалярного произведения $ab = \sum_{i=1}^{N} a_i b_i$ двух векторов большой длины на векторном процессоре. Если компоненты векторов записаны на двух магнитных лентах, то скорость вычисления скалярного произведения лимитируется прежде всего скоростью считывания с них информации. Даже для лучших отечественных перспективных лент эта скорость пока не превышает 320 кбайт/с, или (что то же) 80 тыс. 32-разрядных чисел в 1 с. Поэтому даже скорость процессора в 160 тыс. умножений в секунду, легко достижимая в настоящее время в тривиальном однопроцессорном варианте, оказывается вполне достаточной для вычисления скалярного произведения с максимально возможной в данных условиях скоростью. Применение же специального векторного процессора ничем помочь не сможет.

Разумеется, если подобная задача появилась на входе системы, то указанное препятствие непреодолимо. Иное дело, если она представляет собой часть большой задачи, в которой компоненты векторов вычисляются, а не просто вводятся извне. В большинстве случаев только в специальных методах программирования и расположения данных в памяти могут использоваться при этом все возможности ускорения, предоставляемые векторным процессором.

Надо сказать, что проблема замедления скорости работы в результате взаимодействия с памятью возникает в случае применения обычных (скалярных) сверхбыстродействующих процессоров. Для согласования скоростей работы ОЗУ со скоростью процессора выработан и успешно применяется целый ряд приемов. Это и ОЗУ с ячейками большой длины (на несколько машинных слов), и блочные ОЗУ, использующие эффект перекрытия (одновременного обмена с несколькими блоками ОЗУ), и, наконец, промежуточная сверхбыстродействующая память (СОЗУ) относительно небольшого объема.

Отметим, что существуют задачи, для ускорения решения которых наиболее целесообразно использовать ЗУ с другими формами доступа. Хорошо известно, например, значение ассоциативной памяти для убыстрения решения задач с выборкой информации не по адресу, а по тем или иным признакам. Особенно эффективны ЗУ, которые объединяют в себе возможности как ассоциативного, так и обычного произвольного адресного доступа. Использование такого ОЗУ в американской ЭВМ STARAN позволило получить для задач ПВО и управления воздушным движением производительность порядка 500 млн операций/с.

Переход от ферритовой памяти к полупроводниковой сделал подобные ЗУ с комбинированным доступом относительно недорогими. Это же относится к иного вида памяти — стековой, которая оказывается незаменимой во многих управляющих программах и программах для обработки буквенной информации.

В Институте кибернетики АН УССР работы по так называемой R-технологии программирования привели к понятию R-машины, в которой многие сложные программы упрощаются на 1,5—2 порядка с соответствующим возможным увеличением производительности. Для ее реализации наряду с уже упомянутыми потребуются некоторые дополнительные виды ЗУ, например табличные и так называемые вагонные.

Построение дешевых ЗУ большого объема, в которых реализовались бы одновременно все перечисленные виды доступа, продолжает оставаться достаточно трудной технологической задачей. Для ее решения, возможно, потребуется кроме традиционно применяющихся в вычислительной технике использовать некоторые новые физические принципы.

Во всех рассмотренных случаях распараллеливание касалось исполнения отдельных команд. Сам поток команд в рамках одного процесса носит строго последовательный характер (один поток команд). В действительности же многие процессы (программы) имеют участки, которые могут исполняться параллельно. Такое распараллеливание процессов требует для своей реализации вычислительных систем со многими центральными процессорами.

Один из наиболее простых принципов организации подобных многопроцессорных вычислительных систем — системы с общей шиной. К этой шине (приспособленной для быстрой передачи информации) подсоединяются центральные процессоры и блоки ОЗУ (а также каналы связи с периферийным оборудованием).

В языки программирования таких систем вводятся средства для описания ветвления и слияния процессов, а в операционную систему — средства для синхронизации «стыков» параллельных ветвей и разрешения конфликтов при перекрытиях (во времени), так называемых критических интервалов этих ветвей. Смысл термина «критический» в этом случае заключается в том, что на таких интервалах происходит обращение ветвей процесса к общим ресурсам (блокам ОЗУ, каналам и т. п.).

Ведение программ управления (ОС) в системах с общей шиной поручают одному из центральных процессоров. Подобная централизация управления в случае сложноветвящихся процессов с большим числом параллельных ветвей, сложными системами конфликтов налагает непосильные требования на «центрального управляющего». Серьезные ограничения на общую производительность системы накладывает также ограниченная пропускная способность общей шины.

В результате принцип мультипроцессирования с общей шиной в чистом виде для большинства задач приводит к ограничению числа процессоров в системе до 8—10. После превышения этого порога рост производительности системы резко замедляется, а использование оборудования существенно ухудшается. Данное положение можно получить за счет частичного распараллеливания функций управления (аппаратной поддержки ОС), введения в состав процессоров буферных СОЗУ, увеличения числа шин и других мер.

Более радикальное решение вопроса заключается в более глубоком распараллеливании управления. Одна из базовых идей в этом направлении — организация так называемого потокового управления. Смысл его прежде всего состоит в использовании специального языка, ориентированного на параллельную обработку. Операторы (команды) и данные в таких языках представляются как бы «россыпью», что сопровождается специальными признаками (тегами), позволяющими данным находить свои операторы и свои устройства. Управление в потоковых системах организуется так, чтобы в соответствии с наличным ресурсом рабочих процессоров оперативно соединять в них данные с соответствующими операторами для осуществления необходимых преобразований. Процесс управления оказывается тем самым асинхронным и в максимально возможной степени децентрализованным.

Потоковое управление в принципе можно реализовать в структурах, близких к традиционным системам с общими шинами, к которым наряду с

рабочими процессорами, блоками ОЗУ и каналами интерфейса с периферией добавляются специальные управляющие процессоры, создающие особую «зону управления». Таким образом предлагают строить вычислительную систему (МАРС) Г. И. Марчук и В. Е. Котов (1978).

Был построен образец системы с потоковым управлением и специальным языком параллельного программирования VAL (Valueoriented algorithmic language).

Отметим, что языки параллельного программирования, будучи достаточно удобными для задач моделирования сложных систем, состоящих из большого числа относительно независимых объектов (например, социальных или экономических систем), довольно плохо приспособлены для описания большинства обычных вычислительных задач, имеющих рекурсивную природу. Это обстоятельство, а также необходимость использования уже накопленных библиотек алгоритмов требуют создания сложных трансляторов, переводящих традиционные описания алгоритмов на языки параллельного программирования. Операционные системы многопроцессорных структур с потоковым управлением представляются также достаточно сложными. В этом одна из причин того, что пока сделаны лишь первые шаги к промышленной реализации таких структур, несмотря на то, что в однопроцессорном варианте управление потоком данных (в так называемых системах реального времени) нашло широкое применение уже сравнительно давно.

Дальнейший шаг в развитии структур с потоковым управлением — получение рекурсивных структур, адаптирующихся к реализуемым на них процессам. Основной принцип данного процесса состоит в том, что в системе процессоров (снабженных собственными ОЗУ) с гибкой системой коммутации происходит последовательный захват процессоров реализуемой программой с образованием соответствующих структуре программы связей между ними. В частности, вместо рекурсивного вызова процедур происходит образование новых ветвей взаимосвязанных процессоров. Этот принцип впервые был предложен В. М. Глушковым, М. Б. Игнатьевым, В. А. Мясниковым, В. А. Торгашевым на конгрессе ИФИП (Международная федерация по обработке информации) (Стокгольм, 1974 г.).

Для организации связей в рекурсивной ЭВМ В. А. Мясников и другие предложили иерархическую схему, использующую, по сути, принцип коммутации, применяемый в телефонных сетях. Подобная идея коммутации процессоров (П) с помощью системы коммутационных блоков (К) позволяет в принципе осуществлять коммутацию любой пары процессоров.

При достаточно большом числе параллельно коммутируемых каналов существует возможность получить достаточно сложные структуры связей между процессорами. Как и следовало ожидать, реализация идеи рекурсивной ЭВМ в чистом виде (как и почти всякой чистой концепции) оказалась практически невозможной. Главная трудность здесь состоит в эффективном наложении структуры связей решаемой на ЭВМ задачи на структуру связей процессоров. Сравнительно просто такая задача решается лишь в случае относительной неизменности этой структуры на протяжении решения всей задачи, возможности размещения всей информации (программы и данных) в распределении памяти системы (ОЗУ рабочих процессоров) и игнорировании проблемы начального заполнения системы. Необходимо также принять во внимание неизбежные потери скорости при осуществлении обменов через несколько коммутационных блоков.

В 1978 г. автор данной работы предложил идею усовершенствованной системы коммутации, представляющей наряду с коммутацией и функции

управления. Способность такой системы выполнять функции управления позволяет динамически перестраивать связи между процессорами во время решения задачи.

Вторая базовая идея автора состоит в том, чтобы, используя специальный вариант проблемно-ориентированного структурированного языка, применять управляющие процессоры для предварительного анализа последовательных гнезд циклов структурированной программы. В результате этого анализа с помощью нового математического аппарата (алгебры структур данных) иерархия управляющих процессоров осуществляет такое распределение данных для каждого очередного гнезда циклов между ОЗУ рабочих процессоров и дополнительными ЗУ, подсоединенными через периферийные процессоры, чтобы в максимально возможной степени минимизировать обмены, используя систему иерархической коммутации. При этом для подавляющего большинства задач удается получить прогрессивное уменьшение скоростей обмена по мере повышения уровня иерархии.

Данный результат имеет принципиальное значение, поскольку позволяет устранить опасность снижения быстродействия системы за счет дальних связей. Кроме того, освобождая архитектурные проблемы от сложных конструктивно-радиотехнических вопросов, связанных с высокочастотной связью, он значительно облегчает наладку системы и делает возможным практически неограниченно наращивать уровень иерархии (вплоть до выхода на уровень сетей территориально разнесенных вычислительных систем).

Третья идея состоит в том, чтобы в иерархическом управлении вычислительным процессом использовать все уровни описания программного продукта, полученные в процессе его проектирования и кодирования. С использованием предложенной в начале данной работы производственной аналогии можно сказать, что современные системы, реализующие на всех уровнях управления лишь окончательное машинное представление программного продукта, уподобляются администрации завода, которая осуществляет управление на основе первичной (самой подробной) технологической документации без всяких вторичных (обобщающих) документов. Разумеется, при таком подходе технология программирования должна в максимальной степени соответствовать принципам архитектуры систем, на которых предполагается реализация готовящегося с ее помощью программного продукта.

Описанные принципы позволяют организовать своеобразный «макроконвейер», направленный на распараллеливание циклов и гнезд циклов. Распараллеливания линейных участков программы эта идея в чистом виде, вообще, не предусматривает. Поскольку каждый такой участок в структурированной программе выполняется лишь один раз, то потерей времени на их «однопроцессорное» прохождение в сложных задачах можно всегда пренебречь, так как основные затраты времени приходятся именно на реализацию циклов. Впрочем, как уже отмечалось выше, любой из изложенных в данной работе принципов может комбинироваться с другими, так что при желании можно обеспечить распараллеливание и линейных участков программ (это может оказаться необходимым в некоторых задачах обработки данных в реальном масштабе времени).

Отметим, что предлагаемые архитектура и принципы управления предполагают «мягкую» архитектуру рабочих процессоров. Тем самым в задачу упреждающего планирования прохождения гнезд циклов входит планирование настройки на нужные операторы привлекаемых к вычислениям рабочих процессоров.

Относительно методов решения задач отметим еще, что рассматриваемый метод распараллеливания циклов оказывается эффективным не во всех случаях. Трудным случаем является, например, классический метод решения краевой задачи для обыкновенных дифференциальных уравнений путем «пристрелки». Однако стоит лишь изменить метод — и задача распараллеливается элементарно. При этом на каждой итерации приближение решения точно удовлетворяет краевым условиям, но не удовлетворяет исходным дифференциальным уравнениям. Разбивая очередную траекторию (приближение решения) системой параллельных гиперплоскостей на участки, можем вычислить векторную невязку для каждого такого участка (с точки зрения отличия ее от подходящего отрезка решения системы). Смещая в соответствии с этой невязкой (на разные расстояния) точки данного участка траектории, получаем новое приближение искомой траектории. Этот процесс можно построить так, что он будет сходиться к истинному решению задачи.

Таким образом, в полной мере возможности нового макроконвейерного инструмента можно будет использовать при соответствующем развитии методов решения задач. Проведенные исследования показывают, что для большинства трудных задач, которые встречаются в различных областях науки и техники, метод макроконвейера работает без существенных переделок метода их решения.

В Институте кибернетики АН УССР развит эффективный метод автоматизации формальных преобразований алгоритмов и программ, позволяющий, в частности, во многих случаях получать резкое их убыстрение, не вникая в содержательный смысл решаемых задач. Сегодня таким образом реально улучшаются алгоритмы, запись которых на языке ФОРТРАН содержит до 1500 операторов. С помощью системы автоматизации проектирования схем ЭВМ (системы ПРОЕКТ) эти усовершенствованные алгоритмы реализуются в соответствующей аппаратуре. Тем самым открываются большие дополнительные возможности нахождения новых аппаратных решений для ускорения быстродействия выполнения сложных операторов в универсальных «интеллектуальных» процессорах и целых программ в функционально-ориентированных процессорах.

(Препр. Ин-та кибернетики АН УССР
№ 78—68, 1978)

Сетью ЭВМ принято называть множество ЭВМ, объединенных между собой сетью электрической связи и специальной системой математического обеспечения, позволяющими отдельным ЭВМ и реализующимся на них процессам (программам, обрабатывающим ту или иную информацию) обмениваться друг с другом различными сообщениями.

Для организации связи между удаленными друг от друга ЭВМ определяющее значение имеет то, что принятая форма (как потенциальная, так и импульсная) представления сигналов внутри современных ЭВМ предполагает большую крутизну их фронтов. В результате даже при невысоких темпах посылки сигналов в их естественном (машинном) виде линия связи должна иметь весьма широкую полосу пропускания. Поэтому при прямых связях (без специальной аппаратуры усиления и восстановления сигналов) для передачи машинных сигналов требуются дорогостоящие высокочастотные кабели. Но даже по таким кабелям дальность надежной прямой связи между ЭВМ обычно не превышает нескольких сотен метров.

Используя на приемном конце специальную аппаратуру распознавания, усиления и восстановления формы сигналов, удается увеличить длину прямых связей. Так, разработанная в Институте кибернетики АН УССР в 1975—1977 гг. аппаратура «Барс» позволяет осуществлять устойчивую передачу машинных сигналов по простым изолированным проводам (без использования специальных высокочастотных кабелей) при частоте следования посылок 20—30 кГц на расстояние до 20 км.

Для передачи машинных сигналов на большие расстояния сегодня принято использовать существующие сети телеграфной связи. Для обеспечения передачи информации по относительно узкополосным каналам в этих сетях применяется несколько иная (и отличная от принятых в ЭВМ) форма представления сигналов. Поэтому при использовании этих каналов для передачи машинных сигналов необходимо на их концах (подсоединенных к ЭВМ или терминалам) помещать специальные устройства для преобразования формы сигналов — так называемые модемы. Для понимания возникающих здесь проблем необходимо хотя бы в общих чертах познакомиться с принципами организации современной системы местной и дальней телефонной связи.

Прежде всего оказывается, что для довольно качественной передачи обычной человеческой речи достаточна полоса пропускания в 3 кГц (точнее, диапазон частот от 300 до 3400 Гц). Этот стандарт частоты был назван тональной частотой. Отметим, что он выбран на основании частотных характеристик оконечных устройств (органов речи и слуха человека), а не каких-либо характеристик самой сети связи.

Сигналы, которые передаются в местных сетях при телефонных разговорах, представляют собой простые электрические копии соответствующих звуковых сигналов. Телефонные аппараты подсоединяются к местным

телефонным станциям с помощью так называемых абонентских линий, представляемых физически в виде жил многожильного кабеля. Телефонная станция осуществляет коммутацию абонентских линий, позволяя каждого из подключенных к ней абонентов подсоединить к любому другому. Кроме того, определенная часть коммутируемых кабельных линий предназначается для связи данной телефонной станции с другими станциями, составляющими местную телефонную сеть.

Для дальней (междугородной) телефонной связи организация использования кабелей аналогична, при этом одна физическая линия применяется для одновременной передачи многих телефонных разговоров. С этой целью тональной частотой модулируются высокочастотные сигналы, разнесенные по спектру частот так, чтобы возникающие в результате полосы частот не перекрывали друг друга. Если все эти сигналы передаются по одной физической линии одновременно, то с помощью специальных фильтров на приемном конце их можно разделить и демодулировать в исходную тональную частоту. Для надежного разделения несущие частоты должны быть разнесены по спектру с промежутками, несколько превышающими тональную частоту (в качестве такого промежутка у нас выбрана частота в 4 кГц).

В зависимости от полосы той или иной физической линии связи используются различные степени уплотнения исходного канала (линии связи), т. е. количество виртуальных телефонных каналов, каждый из которых может быть использован (одновременно с другими виртуальными каналами) для передачи телефонного разговора по данной физической линии.

Принятый у нас стандарт предусматривает несколько групп уплотнения. Первичная 12-канальная группа охватывает полосу частот от 60 до 108 кГц. Пять первичных групп составляют вторичную 60-канальную группу с полосой частот от 312 до 552 кГц. Пять таких групп объединяются в 300-канальную группу с полосой частот от 812 до 2044 кГц и т. д.

Поскольку для телеграфной связи требуется меньшая полоса пропускания (в принятом у нас стандарте — 140 Гц), то для улучшения использования линий при передаче телеграфных сообщений применяется аппаратура вторичного уплотнения, образующая в одном телефонном канале 12 телеграфных каналов.

В качестве физических линий при дальней связи, кроме устаревшей проводной связи, используются высокочастотные кабельные линии и радиосвязь в УКВ дециметровом и сантиметровом диапазонах. Поскольку в этих диапазонах устойчивая связь обеспечивается лишь в пределах прямой видимости (без огибания радиоволнами поверхности Земли), то линия связи представляет собой ряд вышек с прямопередающими в виде направленных зеркал антеннами, каждая из которых снабжается аппаратурой для необходимого усиления передаваемых сигналов (так называемые радиорелейные связи). Такими же усилительными станциями (расположенными на определенном расстоянии друг от друга) снабжаются проводные и кабельные линии дальней связи.

В последнее время все большее распространение получают космические (спутниковые) линии связи, использующие в качестве ретрансляционной станции высоколетящие спутники. Особенно удобны так называемые стационарные спутники, период обращения которых вокруг Земли равен периоду обращения Земли вокруг своей оси, а плоскость орбиты совпадает с плоскостью экватора. Такой спутник будет все время висеть над одной и той же точкой земного экватора, не меняя своего положения относительно поверхности Земли.

Спутниковая связь обладает двумя большими преимуществами. Во-первых, она уменьшает количество промежуточных ретрансляционных станций (которые устанавливаются на спутниках) до одной или в крайнем случае до двух. Во-вторых, в спутниковой связи передаваемый сигнал только малую часть расстояния проходит в непосредственной близости к поверхности Земли, в зоне, насыщенной радиопомехами от гроз и разнообразных технических источников (например, систем зажигания автомобильных двигателей). Вследствие этого надежность спутниковой связи оказывается более высокой по сравнению с другими системами связи.

Как уже отмечалось выше, стандарты, принятые для телефонных каналов, определяются частотными характеристиками оконечных устройств (органов речи и слуха человека). Это же наблюдается для телеграфной связи. Казалось бы естественным, чтобы при организации связи между ЭВМ использовался тот же принцип. Тогда задача системы связи сводилась бы к созданию временных каналов соответствующей пропускной способности, чтобы соединяемые ЭВМ могли обмениваться информацией в естественном для них темпе. Создание сети ЭВМ, удаленных друг от друга на любое расстояние, в принципе почти не отличалось бы от создания комплекса ЭВМ, расположенных в пределах прямых связей. Единственным их отличием были бы необходимость модулирования и демодулирования несущих частот системы дальней связи машинными сигналами и дополнительные требования по борьбе с возникающими в линиях связи помехами.

В действительности развитие сетей ЭВМ пошло по линии вписывания системы передачи данных между машинами в уже существующие стандарты сетей телефонной и телеграфной связи. Поэтому возникает необходимость создать многоуровневую систему преобразования потоков данных, которыми обмениваются ЭВМ через сеть связи, и соответствующих систем стандартов представления информации и управления для каждого уровня — так называемых протоколов.

Задачей самого нижнего, так называемого физического, уровня является преобразование стандартных однобитовых сигналов (каждый из которых принимает лишь два значения: 0 и 1) в различимые друг от друга сигналы тональной частоты, а также обратное преобразование. Эта задача решается специальными устройствами модуляции и демодуляции — модемами. Простейшие модемы с частотной модуляцией используют для представления 0 и 1 две различные частоты в пределах полосы 300—3400 Гц. В других типах модемов используется амплитудная или фазовая модуляция. Возможны также и более сложные комбинированные системы.

Скорость передачи дискретной информации с помощью модемов разных типов определяется прежде всего используемым каналом связи. Различают низкоскоростные модемы, использующие телеграфные каналы, и среднескоростные, использующие первичные 12-канальные группы. Внутри этих групп установлены стандарты скорости передачи, зависящие от уровня помех в каналах связи и применяемого метода модуляции.

Для низкоскоростной передачи приняты стандарты 50, 100 и 200 (бит/с), для среднескоростной — 600, 1200, 1800, 2400, 3000, 3600 и 4800 для высокоскоростной — 19,6; 48 кбод и выше. Для капалов с уровнем помех возможно дополнительное увеличение скорости данных, приходящейся на 1 Гц полосы капала. Так, для низкс стандартного телефонного канала (с полосой 3,1 кГц) удается данные со скоростью 9600 бит/с. Для понимания специфики ния информации в сетях ЭВМ необходимо хотя бы в общи

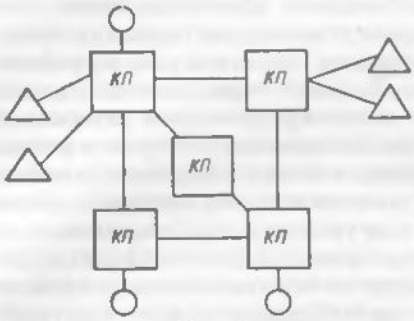


Рис. 1.

комиться с принципами организации (или, как принято говорить, архитектурой) подобных сетей.

Пример сети достаточно простой архитектуры показан на рис. 1. В узлах этой сети (показаны квадратами) расположены так называемые коммутационные процессоры (КП), соединяемые между собой (через модемы) линиями дальней связи. Кружками обозначены ЭВМ, а треугольниками — терминалы пользователей. Они подсоединяются к территориально близким к ним узлам сети (через модемы или в пределах прямой связи без них) с помощью местных абонентских линий или (с модемами) через телефонную сеть.

Коммутационные процессоры могут осуществлять функцию коммутации каналов между абонентами сети (ЭВМ и терминалами), организуя на все время работы двух абонентов постоянный канал связи между ними. Однако подобный способ связи обычно приводит к весьма неэкономному использованию каналов связи. Это связано с тем, что во время сеанса связи между отдельными сообщениями, которыми обмениваются абоненты, могут существовать промежутки ожидания, когда канал, будучи занятым, не используется для фактической передачи информации. Даже при непрерывной передаче информации канал может оказаться недогруженным вследствие малой скорости передачи (что, как правило, происходит при передаче данных с терминалов).

Для более полной загрузки каналов применяется иной способ коммутации — так называемая коммутация сообщений. Суть его в том, что сообщения, которыми обмениваются абоненты сети, могут временно запоминаться в узлах сети (в памяти коммутационных процессоров) и передаваться от узла к узлу наиболее экономным способом (как правило, на большой скорости), обеспечивая гораздо лучшее (по сравнению со способом коммутации каналов) использование дорогостоящих каналов дальней связи.

Однако коммутация произвольных сообщений имеет и свою отрицательную сторону: при варьировании длины сообщений в широких пределах память КП приходится рассчитывать на самые длинные из них. В результате происходит неоправданное увеличение стоимости памяти КП при малом среднем коэффициенте ее загрузки.

Для устранения этого недостатка в современных сетях используется разбивка сообщений на пакеты стандартной длины, передаваемые независимо друг от друга и часто даже по различным путям. Тем самым в задачу управления сетью добавляется управление разборкой и сборкой сообщений, оптимизация маршрутов их передачи в зависимости от загрузки каналов и узлов сети и многое другое.

Опишем основные задачи управления и стандарты представления информации (протоколы) на различных уровнях, начиная с нижнего — так называемого протокола управления информационным каналом (ИКУП). Этот протокол частично может реализоваться в КП, но большей частью — в специальных устройствах, называемых адаптерами линии передачи данных (ЛПД).

Главная задача ИКУП — обеспечение достоверности передачи информации даже при недостаточно надежном канале связи. Эту задачу

можно решить многими способами. Один из них — использование кодов с обнаружением и исправлением ошибок. Простейший из таких кодов — код с контролем по четности, который используется и в ЭВМ. Смысл его в том, что к каждой посылке из заданного числа n битов (в ЭВМ обычно $n = 8$) добавляется еще один двоичный сигнал (0 или 1) так, чтобы общее число единиц в полученной $(n + 1)$-й битовой посылке было четным. Задачей ИКУП в этом случае будут контроль по четности на приемном конце и посылка на предыдущий конец ответного сигнала о результате этой проверки. Если проверка подтвердила правильность приема, то передается следующая посылка, если нет — осуществляется повторная передача неправильно принятой посылки. Именно так выполняется контроль правильности передачи информации между центральной и периферийной частями ЭВМ. В сетях связи такой простой способ не приемлем, так как для сетей связи наиболее характерны так называемые групповые помехи, которые искажают несколько битов посылки подряд.

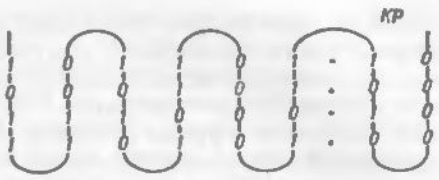


Рис. 2.

Для обнаружения, а тем более исправления ошибок в случае групповых помех обычные способы потребовали бы неоправданно большого числа контрольных разрядов. Поэтому способ контроля видоизменяется в соответствии с идеей, иллюстрируемой на рис. 2, где $КР$ — контрольные разряды, позволяющие обнаружить и исправлять ошибки в каждой строке. Однако передача происходит не по строкам, а по столбцам.

Если длина групповой помехи не превосходит одного столбца, а вероятность появления помехи достаточно мала, то даже при наличии такой помехи, как правило, будет искажаться не более одного разряда в каждой строке, что позволяет на приемном конце не только обнаруживать, но и исправлять ошибки при относительно небольшом числе контрольных разрядов.

Необходимость переупорядочения порядка следования сигналов в посылках до и после передачи сильно усложняет ИКУП. Поэтому на практике обычно употребляют более простой способ контроля правильности передачи посылки: передав посылку от пункта A пункту B, немедленно передают ее обратно пункту A, где сравнивают с запомненной копией переданной посылки. В случае их совпадения в B передается сигнал правильности передачи, и начинается передача следующей посылки. В противном случае процесс передачи посылки повторяется столько раз, сколько потребуется для правильной передачи. Оптимальная длина посылки (обеспечивающая максимум пропускной способности канала) зависит от вероятности искажения передаваемых сигналов. Если эта вероятность равна 10^{-2} (весьма ненадежный канал), то оптимальная длина посылки равна 5 бит. При этом за счет необходимости многократных повторений передачи посылок истинная пропускная способность канала составит всего 1,4 % номинальной. При вероятности ошибки 10^{-4} (канал средней надежности) оптимальная длина посылки равна 57 бит, а коэффициент использования канала поднимается примерно до 50 %. Для каналов высокой надежности (вероятность ошибки 10^{-6}) соответствующие величины будут равны 610 бит и 75 %. Важно подчеркнуть, что правильно построенный ИКУП (и соответствующая аппаратура передачи данных) позволяет осуществлять надежную передачу данных (хотя и с пониженной скоростью) по сколь угодно плохим каналам.

Еще один вопрос, связанный с ИКУП: как отличать одну посылку от другой? Один из широко употребляющих способов решения этой проблемы заключается в использовании специальных маркеров в начале и в конце каждой посылки. Например, маркером может служить комбинация 01111110, т. е. группа из шести последовательных единиц, обрамленных нулями. Чтобы исключить возможность случайного появления такой комбинации в теле посылки, аппаратура передачи данных производит автоматическую вставку (на передающем конце) и исключение (на приемном конце) в теле посылки символа 0 после каждой очередной группы из пяти символов. Тем самым исключается возможность появления группы из шести единиц подряд.

Следующий (за ИКУП) уровень управления передачей информации в сетях ЭВМ регламентируется так называемым протоколом пакетной коммутации и соответствующим математическим обеспечением, встроенным в КП сети. Протокол определяет длину пакета (обычно от 256 до 4096 бит), а также содержание и формат заголовка пакета. В заголовок пакета обязательно включаются адреса абонентов, обменивающихся пакетом. КП, анализируя эти заголовки, определяет, какому из узлов сети (соседнему с данным) переслать пакет. При этом часть операционной системы сети (ОСС), встроенная в КП, решает задачу оптимизации движения пакетов от узла к узлу так, чтобы обеспечить, с одной стороны, наилучшую загрузку линий связи и самих КП, а с другой — наиболее быструю передачу пакетов конечным адресатам. Для точного решения этой задачи необходим специальный управляющий процессор, непрерывно получающий информацию о состоянии всех узлов сети и линий связи.

Такое централизованное управление загружает сеть передачей большого количества служебной информации и предъявляет повышенные требования к надежности управляющего процессора. Поэтому на практике обычно используют упрощенные приемы приближенной локальной оптимизации, когда решение о направлении пересылки вырабатывается КП, в котором этот пакет находится, на основании информации о непосредственно прилегающих к нему участках сети.

Еще одна задача, решаемая на этом уровне,— идентификация и аннулирование дублей пакетов, которые могут возникнуть в результате множественности маршрутов и возможности зацикливания. С этой целью в заголовке пакетв должен содержаться идентификатор породившего его сообщения. Обычно он составляется из сетевого номера пославшей сообщение ЭВМ, номера процесса в этой ЭВМ (а иногда и кода пользователя), инициировавшего данное сообщение, и, наконец, идентификатора самого сообщения в этом процессе.

На следующем уровне управления встречаемся с задачей сборки и разборки сообщений на отдельные пакеты с формированием и последующим уничтожением соответствующих заголовков пакетов. Этот уровень управления регламентируется так называемым транспортным протоколом и реализуется частью ОСС, называемой обычно процессором абонентских сообщений (ПАС). В первых сетях ПАС обычно встраивался в КП. В более поздних сетях (например, во французской cyclades) ПАС реализуется в абонентских ЭВМ (АВМ) и управляется их операционными системами как одна из задач пользователя. Такое решение обеспечивает большую стандартность сети и ее независимость от типа подключаемых к ней АВМ. При подсоединении АВМ к КП стандартная часть интерфейса реализуется в КП, а нестандартная — в самой АВМ и специальных сетевых адаптерах.

Еще более высокий уровень управления (реализующийся в АВМ) организует потоки сообщений от процесса к процессу, включая задачу формирования заголовков сообщений на передающем конце и их упорядочение в правильный поток (упорядоченный по номерам сообщений). На этом уровне реализуются макрокоманды создания и уничтожения так называемых виртуальных каналов, связывающих между собой удаленные процессы. Соответствующая часть ОСС (встраиваемая в АВМ) называется супервизором виртуальных каналов.

Дальнейшая детализация управления решается на уровне протокола процесс — процесс и соответствующей части ОСС (встроенной в АВМ) так называемым сетевым супервизором процессов (СПС). На этом уровне организуется связь портов. В качестве порта могут выступать некоторый массив данных (файл), устройство ввода — вывода, некоторая точка программы, в которой генерируются или потребляются сообщения. Это примеры индивидуальных портов (принадлежащих определенному процессу); примером коллективных портов могут служить входы в системы управления базами данных (СУБД) с последующим уточнением адресата (файла или записи) с помощью средств самих СУБД.

Каждый порт снабжается сетевым идентификатором, состоящим из сетевого идентификатора АВМ, идентификатора пользователя этой АВМ, идентификатора, присвоенного процессу, и, наконец, идентификатора порта в процессе. Порт снабжается необходимой буферной памятью для размещения формируемых и принимаемых сообщений. На уровне протокола процесс — процесс реализуются макрокоманды (связанные с ОС АВМ): «открыть порт», «закрыть порт», «соединить (или разъединить) порт с портом» (создать или аннулировать соответствующий виртуальный канал).

Еще более высоким уровнем ОСС и соответствующих протоколов является уровень управления решением задач на сети. Основная решаемая здесь задача — создание виртуальных сетей процессов (в различных АВМ). Обычно этот уровень реализуется в виде совокупности мониторов сетевых абонентских служб, каждый из которых предназначен для исполнения сетью той или иной определенной функции (например, справочно-информационной). Такие мониторы могут встраиваться в АВМ или в специальные информационно-диспетчерские ЭВМ (ИДВМ). В таких ИДВМ может, например, реализоваться функция безадресного обращения любого пользователя сети к распределенному (между абонентскими ЭВМ) банку данных через соответствующую систему сетевых каталогов. В задачу мониторов абонентских служб входит планирование и управление процессом решения задач на сети с учетом загрузки отдельных АВМ и линий связи. Все эти функции регламентируются специальными протоколами.

Помимо уже перечисленных в ряде действующих сетей используются и другие протоколы, например специальные протоколы для пересылки файлов, для удаленного ввода заданий, протокол виртуального терминала и др.

В связи с развитием многих сетей национального и даже межнационального масштаба с различными системами протоколов возникает специальная задача организации межсетевого взаимодействия. Для решения этой задачи обычно устанавливается общий (межсетевой) протокол процесс — процесс, фиксируются единый формат сообщений и единый язык управления обменом сообщениями. Для преобразования подобных межсетевых сообщений в форматы, принятые в отдельных сетях, некоторые узлы (КП) объединяемых сетей соединяются друг с другом линиями связи

через специальные устройства (обычно универсальные ЭВМ с адапте рами), называемые шлюзами.

Идеи создания универсальных (для решения широкого круга вычис лительных и справочно-информационных задач) сетей ЭВМ возникли в кон це 50-х — начале 60-х годов. В 1963 г. под руководством автора был раз работан эскизный проект Единой государственной сети вычислительных центров (ЕГСВЦ), предусматривавший объединение в единую систему нескольких десятков тысяч ЭВМ и сотен тысяч терминалов, охватываю щих всю территорию страны. В этом проекте предусматривалось без адресное обращение к распределенным банкам данных, решение сложных задач на сотнях удаленных друг от друга ЭВМ в диалоговом режиме и мно гое другое.

Практическая реализация проекта началась в 70-е годы. За рубежом были созданы хотя и меньших масштабов, но тем не менее эффективно ра ботающие универсальные сети (сети ARPA и CYBERNET в США (1969 г.) и др.).

Отметим, однако, что некоторые идеи проекта ЕГСВЦ до сих пор не нашли полного решения. Например, это касается идеи иерархической структуры сети, верхний уровень которой должны составлять сверхмощ ные общегосударственные ВЦ с широкополосными связями (в обход обыч ной канало образующей аппаратуры) и соответствующим резким упроще нием протоколов обменов на этом уровне.

(USSR, computing in.— In: Encyclopedia
of computer science and technology.
New York and Basel, M. Dekker.— 1979.— 13.— P. 498—507).

Развитие цифровой электронной вычислительной техники в СССР началось в 1947—1948 гг., когда академик С. А. Лебедев в Институте электротехники АН УССР начал работу по созданию первого в СССР электронного компьютера «МЭСМ». В 1950 г. заработал макет «МЭСМ», а в 1951 г. компьютер был официально введен в эксплуатацию и на нем началось регулярное решение задач. Машина «МЭСМ» была построена по трехадресному принципу с быстродействием 50 операций/с. Она оперировала с 20-разрядными двоичными числами и имела оперативную память на электронных лампах объемом в 100 ячеек. Позднее к «МЭСМ» был подключен магнитный барабан.

Параллельно с этим в Москве в Институте точной механики и вычислительной техники С. А. Лебедев организовал разработку гораздо более мощного компьютера «БЭСМ» со средним быстродействием около 10 тыс. трехадресных операций в секунду над 39-разрядными словами. Оперативная память на электронно-акустических линиях задержки емкостью в 1024 слова вскоре была заменена памятью на электронно-лучевых трубках, а затем — на ферритовых сердечниках. Внешнее ЗУ было представлено двумя магнитными барабанами и магнитной лентой емкостью более 100 тыс. слов. «БЭСМ» была введена в эксплуатацию в 1952—1953 гг. и послужила основой технической базы созданного в 1955 г. Вычислительного центра АН СССР, возглавляемого со времени его основания академиком А. А. Дородницыным. Одновременно с «БЭСМ» в Москве велась разработка (под руководством Ю. Я. Базилевского) компьютера «Стрела» с меньшим быстродействием, чем «БЭСМ», но с большим объемом оперативной памяти. Разработка «Стрелы» была закончена в 1953 г. Машины «БЭСМ» и «Стрела» выпускались серийно.

В 50-е годы коллективы разработчиков ЭВМ сложились и в ряде других городов. В Пензе под руководством Б. И. Рамеева были разработаны и выпускались серийно машины серии «Урал» («Урал-1» — «Урал-4»). В Ереване под руководством Ф. Т. Саркисяна разрабатывались и выпускались компьютеры серии «Раздан». Коллективом ученых Минска (В. В. Пржиялковский и др.) была организована разработка и выпуск машин серии «Минск» («Минск-1», «Минск-11», «Минск-12», «Минск-14»). В Киеве создана ЭВМ «Киев».

В качестве элементной базы у всех машин, создававшихся в 50-е годы, использовались электронные лампы, полупроводниковые диоды и ферритовые сердечники. Самой мощной из компьютеров 50-х годов была машина М-20, разработанная в 1958 г. в Москве под руководством С. А. Лебедева. Среднее быстродействие ее — 20 тыс. трехадресных операций в секунду над 45-разрядными словами (с плавающей запятой), оперативная память на ферритовых сердечниках — объемом в 4096 слов.

Все перечисленные компьютеры первого поколения предназначались в основном для научных расчетов. На их базе, кроме уже упомянутого Вычислительного центра АН СССР, были созданы ВЦ в Академиях наук республик, а также в ряде крупных научно-исследовательских, проектно-конструкторских институтов и университетов.

В это же время начала развиваться компьютерная наука. В 1957—1958 гг. в Киеве был разработан первый универсальный процедурно-ориентированный язык программирования — так называемый адресный язык (В. С. Королюк, Е. Л. Ющенко). В Москве А. А. Ляпунов и его ученики разрабатывали язык операторных схем программ. На базе адресного языка и языка операторных схем в Киеве и Москве были созданы первые системы автоматизации программирования (компайлеры для ЭВМ «Киев», «БЭСМ» и др.).

Начались работы по теории автоматов, искусственному интеллекту и дискретному анализу (В. М. Глушков, Ю. И. Журавлев, О. Б. Лупанов, С. В. Яблонский и др.).

Мощное развитие получил численный анализ, к развитию которого были привлечены крупнейшие советские математики (И. М. Гельфанд, А. А. Дородницын, М. В. Келдыш, М. А. Лаврентьев, А. Н. Тихонов и др.).

В 60-е годы началась эпоха машин второго поколения, строящихся на базе транзисторов. На транзисторной элементной базе развивались все сложившиеся ранее семейства машин. ЭВМ средней мощности для научных расчетов в этот период явились московские разработки («М-220», «БЭСМ-3», «БЭСМ-4»), пензенские («Урал-11», «Урал-14» и несколько позднее «Урал-16»), минские («Минск-22», «Минск-23», Минск-32»), ереванские («Раздан-2», «Раздан-3») и др.

Особо отметим ЭВМ большой мощности для научных расчетов «БЭСМ-6», созданную в 1967 г. в Институте точной механики и вычислительной техники под руководством С. А. Лебедева и В. А. Мельникова. Машина работает с 50-разрядными двоичными словами: использует одно-адресную систему команд; ОЗУ — на ферритовых сердечниках емкостью от 32 до 128 тыс. слов с временем цикла 2 мкс, регистровая память — 16 слов с временем цикла 300 нс. Внешние ЗУ включают 16 магнитных барабанов по 32 тыс. слов и 32 лентопротяжных механизма емкостью свыше миллиона слов на одно устройство. Время сложения двух чисел с плавающей запятой — 1,2 мкс, умножения — 2,1 мкс. ЭВС «БЭСМ-6» имеет развитое математическое обеспечение. Операционная система компьютера организует мультипрограммную обработку нескольких задач, каждая из которых располагает полным объемом виртуальной памяти. Система автоматизации программирования использует в качестве входных языков ФОРТРАН, АЛГОЛ-60, ЛИСП. В разработке математического обеспечения «БЭСМ-6» принимали участие ведущие советские программисты Л. Н. Королев, М. Р. Шура-Бура, Н. Н. Говорун, Э. З. Любимский и др.

В этот период были созданы и другие системы автоматизации программирования для больших и средних ЭВМ, среди которых прежде всего отметим так называемую α-систему для ЭВМ «М-20м», созданную в Новосибирске под руководством А. П. Ершова. В качестве входного языка система использует расширение языка АЛГОЛ-60. Впоследствии система была распространена на ЭВМ «БЭСМ-6». Помимо «БЭСМ-6» в 60-е годы некоторое распространение получили и другие типы высокопроизводительных ЭВМ, например «Весна».

Большое развитие в это время получили также полупроводниковые миникомпьютеры для научных расчетов. Наиболее распространенными были ереванские миникомпьютеры «Наири-1» (1964 г.) и «Наири-2» (1967 г.) (Г. Е. Овсепян и др.), а также киевские миникомпьютеры «Промінь» (1963 г.), «МИР-1» (1965 г.) и «МИР-2» (1969 г.) (В. М. Глушков и др.). В машине «Промінь» впервые реализована двухуровневая асинхронная микропрограммная система управления, получившая дальнейшее развитие в ЭВМ класса «МИР». В ЭВМ «МИР-1» и «МИР-2» была впервые реализована идея повышения уровня «машинного интеллекта» за счет применения развитых внутренних машинных языков и высокоэффективных интерпретирующих систем, построенных на принципах иерархического микропрограммного управления. Для этих компьютеров были созданы специальные входные языки МИР и (его расширение) АНАЛИТИК, в основе совпадающие с внутренними языками машин «МИР-1» и «МИР-2», так что процесс трансляции с входного языка на внутренний практически почти исключен и сводится в основном к операциям перекодирования.

В языке АНАЛИТИК можно оперировать не только целыми числами и десятичными дробями, но и обыкновенными дробями. Операторами языка служат операции аналитического дифференцирования и интегрирования, применения тождеств к выражениям и др. Компьютер «МИР-2» обеспечивает диалог с пользователем через дисплей со световым пером.

Вследствие ступенчатой организации микропрограммного управления и оптимизации микропрограмм удалось добиться двух целей: во-первых, разместить в ограниченном объеме ПЗУ (only read memory) систему интерпретации сложного языка (емкость ПЗУ в компьютере «МИР-2» — около 200 тыс. байт), во-вторых, обеспечить высокую скорость интерпретации. В связи со сказанным «МИР-2» может успешно соревноваться в скорости выполнения аналитических преобразований с компьютерами, превосходящими его по номинальному быстродействию в десятки раз.

В 60-е годы ЭВМ начали широко применяться для управления технологическими процессами, а также для сбора и обработки экспериментальных данных в реальном масштабе времени. В 50-е годы для этих целей использовались лишь специализированные вычислительные устройства. В 1958 г. автор данной статьи высказал идею создания универсальной управляющей ЭВМ, обладающей стандартизированным интерфейсом с аналоговыми устройствами (датчиками с исполнительными механизмами) и операционной системой реального времени. В 1961 г. такая машина под названием «Днепр-1» была создана в Институте кибернетики АН УССР (В. М. Глушков, Б. Н. Малиновский и др.) и с этого года до начала 70-х годов выпускалась промышленностью. «Днепр-1» была первой серийно выпускавшейся советской полупроводниковой универсальной ЭВМ.

В 1967 г. в Институте кибернетики АН УССР была создана новая, более мощная управляющая ЭВМ «Днепр-2», также выпускавшаяся серийно. Машина имела развитую систему прерываний и обеспечивала одновременную работу более чем с 1600 входными и более чем с 1000 выходными аналоговыми устройствами различных классов.

В 1963 г. была разработана и некоторое время выпускалась серийно малогабаритная управляющая машина «УМШ1».

В 60-е годы впервые в достаточно широких масштабах началось применение ЭВМ для планово-экономических расчетов. На многих предприятиях старая счетно-перфорационная техника стала заменяться ЭВМ. Были созданы многие десятки вычислительных центров для коммерческих расчетов. Этот процесс сильно тормозился тем, что отечественная промыш-

ленность в 60-е годы была ориентирована в основном на выпуск ЭВМ для научных расчетов и малогабаритных управляющих ЭВМ. Оборудование для автоматизированных систем управления (АСУ) и систем обработки коммерческих данных с большими объемами информации либо выпускалось в незначительных количествах, либо не выпускалось вовсе. Однако, несмотря на эти трудности, на ряде предприятий удалось уже в эти годы построить эффективные АСУ. В качестве примера можно отметить АСУ на телевизионном заводе во Львове (В. М. Глушков, В. И. Скурихин и др.), созданную вначале на базе скомплексированных ЭВМ «Минск-22», а затем — на комплексе из двух машин «Минск-32», снабженном некоторой дополнительной (специально разработанной для этой цели) аппаратурой. Несмотря на отсутствие в системе магнитных дисков, удалось создать интегрированный банк данных с обновлением в реальном масштабе времени и решать не только учетные, но и планово-управленческие задачи, в частности задачи оперативно-календарного планирования (теории расписаний).

В 1963 г. автором был разработан первый проект объединения АСУ в единую общегосударственную систему — государственную сеть вычислительных центров (В. М. Глушков и др.).

В 70-е годы в результате решений XXIV съезда Коммунистической партии Советского Союза резко увеличились темпы создания АСУ. Начали создаваться основы государственной системы передачи данных. Промышленность была переориентирована на выпуск ЭВМ третьего поколения, пригодных как для научных, так и для коммерческих расчетов. Требования совместимости ЭВМ для работы в сетях и резко увеличившиеся потребности в обмене (в том числе международном) программами и информацией на машинных носителях вызвали необходимость в большей стандартизации и унификации выпускаемых ЭВМ. С этой целью в начале 70-х годов был разработан ряд ЭВМ средней и высокой производительности (ЕС ЭВМ). В разработке и выпуске ЕС ЭВМ кроме СССР приняли участие Болгария, Венгрия, Польша, Чехословакия и Германская Демократическая Республика. По соображениям облегчения международного обмена программами и информацией в ЕС ЭВМ обеспечена совместимость с наиболее распространенными в мире моделями ЭВМ третьего поколения «ИБМ-360» и «ИБМ-370».

Архитектура и программные средства ЕС ЭВМ также близки к решениям, принятым в системах «ИБМ-360» и «ИБМ-370», поэтому нет нужды останавливаться на них подробнее.

Приведем лишь некоторые характеристики одной из старших моделей ряда ЕС 1060: быстродействие этой машины 10^6 операций/с по смеси Гибсона; емкость ОЗУ от 2048 до 8192 кбайт, элементная база (включая быструю память) — интегральные схемы. Отметим, что наиболее производительные ЭВМ для научных расчетов в 70-е годы создавались вне рамок ЕС ЭВМ. При их разработке предпочтение отдавалось совместимости с наиболее распространенной в СССР высокопроизводительной ЭВМ второго поколения «БЭСМ-6».

В области миникомпьютеров в конце 60-х — начале 70-х годов наряду с дальнейшим развитием уже упомянутых выше систем «МИР» и «НАИРИ» («МИР-31», «МИР-32», «НАИРИ-3», «НАИРИ-4») развивались новые семейства ЭВМ, в частности ЭВМ серии АСВТ «М-6000» и «М-7000» (В. В. Резанов и др.). Эти миникомпьютеры нашли широкое применение для управления технологическими процессами. На базе интегральной технологии были созданы компактные (настольные) миникомпьютеры «М-180», «Электроника-100», «Электроника-200» и др.

Появление микропроцессоров привело в 1975—1976 гг. к созданию и промышленному выпуску первых микрокомпьютеров.

К концу первой половины 70-х годов возникла необходимость унифицировать миникомпьютеры, подобно тому как это было сделано в отношении больших и средних машин. В сотрудничестве с учеными Болгарии, Венгрии, Польши, Чехословакии и ГДР был создан ряд миникомпьютеров третьего поколения «СМ1», «СМ2» и «СМ3» (с соответствующей периферией), перекрывающий широкий диапазон применений, начиная от научных расчетов и кончая управлением технологическими процессами и обработкой экспериментальных данных в реальном масштабе времени.

В области технологии изготовления микросхем большой степени интеграции кроме традиционных методов успешно развивалась элионная технология. Для управления электронными и ионными пучками были созданы специализированные миникомпьютеры «Киев-67» и «Киев-70» (В. М. Глушков, В. П. Деркач и др.). Созданные на базе «Киев-70» установки обеспечили точность изготовления микросхем порядка 0,1 мкм. Наряду с электронной литографией были разработаны методы прямого изготовления (p — n)-переходов методами электронно-лучевого легирования (В. П. Деркач и др.).

К началу 70-х годов все ведущие организации, занимающиеся созданием новых ЭВМ, построили у себя автоматизированные системы проектирования печатных плат и междуплатных соединений. Были созданы системы автоматизированного проектирования больших интегральных схем.

Проектирование архитектуры ЭВМ и операционных систем уже в 60-е годы опиралось на системы моделирования. Широкое применение для этих целей нашел разработанный в Институте кибернетики АН УССР в 1966—1968 гг. язык моделирования дискретных систем СЛЭНГ с соответствующими системами компиляции составленных на нем программ (В. М. Глушков, Л. А. Калиниченко, Т. П. Марьянович и др.). В 1974 г. этот же коллектив (без Л. А. Калиниченко) разработал язык и систему моделирования НЕДИС, включающую в себя средства описания и моделирования не только дискретных, но и непрерывных систем.

В области логического проектирования наряду с традиционным математическим аппаратом булевой алгебры и теории автоматов с конца 60-х годов начали применяться более глубокие методы оптимизации микропрограмм и структур ЭВМ. Подобные применения были подготовлены работами по формальным преобразованиям схем программ (Ю. И. Янов, А. П. Ершов, Р. И. Подловченко и др.) и созданием в 1966—1967 гг. нового раздела дискретной математики — теории алгоритмических алгебр и дискретных преобразователей (В. М. Глушков, А. А. Летичевский и др.). Последние результаты позволили представлять программы и микропрограммы ЭВМ как формулы в некоторой алгебре и осуществлять любые их эквивалентные преобразования. На базе этой теории в Институте кибернетики АН УССР создана система автоматизации проектирования ЭВМ вместе с их математическим обеспечением, допускающая в диалоговом режиме глубокую оптимизацию архитектуры и программного обеспечения разрабатываемых ЭВМ (В. М. Глушков, Ю. В. Капитонова и др.). Особенно эффективна эта система в случае, когда приходится проектировать ЭВМ нетрадиционной структуры, ЭВМ серии «МИР». Для этой системы разработан ряд оригинальных проблемно-ориентированных языков.

Автоматизация программирования в 60-е годы наряду с научными задачами стала все в большей мере затрагивать сферу задач обработки данных

в АСУ. Кроме известных языков, применяющихся во всех странах для этих целей (КОБОЛ и др.), в 1964—1966 гг. были разработаны отечественные языки АЛГЭК и АЛГЭМ, являющиеся своеобразными гибридами языков АЛГОЛ-60 и КОБОЛ, а также соответствующие системы компиляции (М. А. Королев, А. И. Китов и др.).

Для систем программирования, созданных в 70-е годы, характерна ориентация на одновременную реализацию нескольких языков, в них используются развитые средства редактирования, отладки и контроля программ. Начали применяться синтаксически управляемые трансляторы, создаются инструментальные системы программирования (А. П. Ершов, С. С. Лавров, Е. Л. Ющенко, М. Р. Шура-Бура, Г. С. Цейтин и др.). Развивалась теория программирования, теория формальных языков, теория структур данных. Изучались проблемы параллельного (асинхронного) программирования, вопросы доказательства утверждений о программах, методы построения полных систем примеров для отладки программ (Я. М. Бардзинь, А. П. Ершов, В. Е. Котов, В. Н. Редько, Е. Л. Ющенко и др.). Дальнейшее развитие получили автоматно-алгебраические методы решения задач анализа и оптимизации алгоритмов. Были найдены подходы к составлению больших программ и систем программ, близкие к идеям структурного программирования (В. М. Глушков, Ю. В. Капитонова, А. А. Летичевский и др.).

В области архитектуры ЭВМ кроме уже отмеченных работ, нашедших воплощение в реальных машинах, в 60-е и 70-е годы проводилась большая теоретическая работа по однородным вычислительным средам (Э. В. Евреинов и др.). Эта работа привела к практическим результатам в виде разработанных методов для параллельных вычислений, которые нашли применение при решении сложных задач на созданных еще в 60-е годы многомашинных вычислительных комплексах и многопроцессорных ЭВМ второй половины 70-х годов. Отметим еще идею рекурсивной организации многопроцессорных вычислительных систем (В. М. Глушков, М. П. Игнатьев, В. А. Мясников, В. А. Торгашев) и работы по созданию квазианалоговых ЭВМ (Г. Е. Пухов и др.).

Для применения ЭВМ в 70-е годы характерно не только дальнейшее развитие вычислительных центров общего назначения, предназначенных для случайных потоков разнообразных задач от широкого круга пользователей. Преимущественную роль в этот период начинают играть специализированные информационно-вычислительные центры, предназначенные для решения задач определенных классов. Это — системы обработки экспериментальных данных, системы для управления сложными технологическими процессами (включая станки с числовым программным управлением), автоматизированные системы организационного управления и обработки данных различных классов, системы комплексной автоматизации испытаний сложных объектов, системы автоматизации проектно-конструкторских работ тех или иных классов и т. п.

Для таких центров характерны более регулярные потоки задач, необходимость комплексирования ЭВМ, а также другого оборудования, в том числе и такого, которое не применяется в ВЦ общего назначения, наличие собственных банков данных коллективного пользования, возможности диалоговых режимов работы и др. Это вызывает необходимость разработки специального математического обеспечения, включая специализированные операционные системы и специализированные системы автоматизации программирования. В 70-е годы подобных специализированных ВЦ насчитывалось сотни и даже тысячи. Начиная с 1967 г. для таких спе-

циализированных ВЦ начали создаваться типовые проектные решения, ориентированные на разные классы применений и разные конфигурации технических средств. Так, в 1967—1971 гг. была разработана типовая система автоматизации организационного управления для машиностроительных и приборостроительных предприятий, применяемая затем на нескольких сотнях заводов (В. М. Глушков и др.).

Для ВЦ общего назначения в 70-е годы начали развиваться системы коллективного пользования с удаленными терминалами (в Киеве, Новосибирске, а затем и в других городах). В специализированных ВЦ удаленные терминалы использовались уже в 60-е годы.

Интенсивными темпами в эти годы развивалась вычислительная математика и соответствующие пакеты прикладных программ. Кроме традиционных задач численного анализа (особенно численных методов решения задач математической физики) (А. А. Дородницын, А. Н. Тихонов, А. А. Самарский, Н. Н. Яненко, Т. М. Энеев и др.) интенсивно развивались методы математического программирования — для нелинейных, стохастических, динамических и целочисленных задач большой размерности (Н. Н. Моисеев, В. С. Михалевич, Ю. М. Ермольев, В. А. Емеличев, Б. Н. Пшеничный, Н. З. Шор и др.).

Получены интересные результаты в области теории и машинных методов распознавания образов (Ю. И. Журавлев, В. А. Ковалевский и др.), в области машинного доказательства теорем (В. М. Глушков, А. А. Летичевский, Ю. В. Капитонова, С. Ю. Маслов, Н. А. Шанин и др.), методов математического моделирования сложных систем (Н. Н. Бусленко, И. Н. Коваленко и др.).

ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ТЕХНИЧЕСКИЙ ПРОГРЕСС

(Научно-технический прогресс
в промышленности Донбасса, 4.1. Донецк.— 1969)

Мы живем в эпоху, которая в техническом плане является революционной. Наука и техника развиваются стремительными темпами и оказывают значительное влияние на развитие производительных сил.

Общий смысл научно-технической революции заключается в том, что наука становится непосредственной производительной силой, поскольку время между научным открытием и его конструкторским, инженерным оформлением и внедрением, которое в XIX в. составляло десятки лет, сократилось до нескольких лет. Поэтому темпы развития народного хозяйства не только в количественном, но и в качественном отношении совершенно не могут быть сравнимы с темпами, наблюдавшимися ранее.

Кибернетику и вычислительную технику справедливо называют центральным звеном этой научно-технической революции. Почему это так?

Кибернетика и вычислительная техника связаны с научно-техническим прогрессом многими способами. Само развитие науки весьма тесно связано с вычислительными машинами, и, как правило, без них оно становится невозможным.

Серьезные проблемы атомной и ракетной техники принципиально были бы неразрешимы, если бы не существовали электронно-вычислительные машины. Многие расчеты, связанные с развитием новой техники, требуют нескольких тысяч и даже десятков тысяч человеко-лет работы и не могут быть принципиально сокращены за счет увеличения количества людей, так как эти расчеты носят последовательный характер.

Некоторые новейшие области науки и техники не могли бы развиваться такими быстрыми темпами, если бы не было электронных машин. Электронно-вычислительные машины начинают использоваться непосредственно в инженерном проектировании, создаются целые системы, принципиально изменяющие наши взгляды на вопросы технического проектирования. Испытания многих сложных приборов новой техники ведутся с помощью вычислительных машин.

Усложнение современного производства приводит к тому, что управление производством без электронных машин, без кибернетики становится невозможным. Смысл управления производством на каждом уровне управления прежде всего заключается в управлении связями между отдельными ячейками.

Как один цех взаимодействует на заводе с другим цехом — это прежде всего дело директора завода. А как взаимодействуют заводы друг с другом, как они связаны с другими отраслями промышленности — это прежде всего дело министерства. Эти взаимоотношения за последние годы особенно усложнились в связи с научно-технической революцией, что вызвало резкое увеличение сложности задач управления нашей экономикой.

Для примера рассмотрим, что представляло собой материально-техническое снабжение любого машиностроительного завода в начале XX в. Завод выпускал машины, которые практически полностью изготовлялись на этом заводе. У него не было поставщиков, кроме поставщиков угля, кокса, чугуна. Современный же машиностроительный завод имеет сотни поставщиков, и изделия, которые он выпускает, включают в себя сложные устройства и узлы. Наименования номенклатуры материально-технического снабжения завода часто исчисляются десятками тысяч. А сложность самого изготовления продукции такова, что перебои с поставками резко сказываются на производительности труда и выпуске продукции.

В 60-е годы по скромным оценкам сложность задач управления нашей промышленностью увеличилась в несколько сотен раз. Если сравнить количество решаемых задач для управления экономикой в народном хозяйстве в 30-е годы и сегодня, то оказывается, что в переводе на производительность управленческого труда по грубым оценкам в 1930—1935 гг. было затрачено 100 млн человеко-лет работы в год по управлению, а в 1966—1967 гг.— 10 млрд человеко-лет. Это означает, что в 1930—1935 гг. можно было создать систему материальной и моральной заинтересованности так, чтобы каждый из 100 млн работающих в нашем хозяйстве думал не только о своем участке работы, но и о соседних участках и вносил свой вклад в управление. Принципиально эти задачи управления в 1930—1935 гг. можно было решать вручную, без использования машин. В 1966—1967 гг. такие задачи решать вручную уже было невозможно, так как для этого потребовалось бы очень много людей. Поэтому основная задача управления экономикой состоит в том, чтобы соединять различные формы экономической реформы с широким использованием средств автоматики, прежде всего электронно-вычислительной техники. Без этого качество управления будет ухудшаться, а следовательно, и потери, которые несет народное хозяйство, будут увеличиваться. Вторая техническая революция — это прежде всего революция в области средств управления.

Первая техническая революция началась с изобретения паровой машины. Основная цель ее — механизация мускульных усилий человека. Это прежде всего была энергетическая революция, она продолжается и сейчас (на смену тепловой пришла электрическая, а затем атомная энергия и т. д.).

Но в настоящее время центром внимания второй научно-технической революции является не столько автоматизация физического труда, сколько автоматизация умственного труда, прежде всего в сфере управления в широком смысле этого слова, потому что инженерно-конструкторская работа, подготовка технологии — это тоже часть управления промышленностью.

Рассмотрим изменения, которые произошли в мире за последние годы в области применения вычислительных машин. Вначале электронно-вычислительные машины применялись эпизодически для решения в основном вычислительных задач. Если инженеру требуется рассчитать конструкцию железобетонного здания, то он в основном делает эту работу вручную, и когда встречается особо сложная задача, где надо много вычислять, тогда он обращается в вычислительный центр, где программирует и вводит данные задачи в вычислительную машину. В настоящее время положение коренным образом изменилось. Начинают создаваться системы автоматизации проектирования. Они отличаются тем, что автоматизируются комплексно все участки проектно-конструкторских работ, чертежные и различного рода вспомогательные работы. Машина дает готовую продукцию в

виде технической документации. В машину закладывается не одна программа, а много сотен программ, рассчитывающих разные участки проекта; происходит обмен данными между программами.

Проект все время находится в памяти машины и выдается уже в готовом виде. Все промежуточные стадии, которые раньше проходили через человека, выполняются внутри машины. Спрашивается, какова же роль человека в таком проектировании? Не умаляется ли здесь роль конструктора и технолога и т. д.? При современном уровне развития техники и кибернетики задача построения полностью автоматических систем по проектно-конструкторским работам для сложных изделий еще нереальна. Поэтому мы говорим не об автоматических системах, а об автоматизированных системах. Автоматизированные системы — это системы, в которых используются лучшие творческие стороны мышления человека и организуется совместная работа людей с вычислительными машинами. В настоящее время Институт кибернетики АН УССР занимается в первую очередь такой системой по автоматизации и проектированию самих вычислительных машин. В данном направлении ученые института были пионерами, заняли ведущие позиции в мире и в настоящее время успешно продолжают работу в данной области.

Электронно-вычислительная машина работает с такой большой скоростью, что одновременно может обслуживать не одного, а нескольких конструкторов по разным проектам. Конструктор пользуется специальным пультом в виде электрифицированной пишущей машинки, с помощью которой он может сообщать некоторые данные машине.

На специальный экран типа телевизионного машина выдает промежуточные чертежи и эскизы. У конструктора имеется специальный световой карандаш, которым он может на этом экране делать пометки. То, что он напишет, остается на экране в виде светящегося кружочка. Работа выполняется таким образом, что в память машины загружается большое количество программ, которые позволяют прежде всего оценивать конструкцию, скажем, жилого дома. Например, нужно сделать планировку этажа. В машину закладываются данные, соответствующие требованиям по планировке квартир, метражу и т. д. Включается программа, выполняющая планировку этажа. После этого определяется максимальная используемая жилая площадь по стоимости квадратного метра. Машина это делает лучше конструктора с точки зрения плотности планировки в заданном объеме и выдает ему соответствующий расчет. Конструктор видит результаты на экране и вносит необходимые коррективы.

Таким образом, в совместной творческой работе конструктора и машины рождается новый проект. Используются лучшие стороны человеческой творческой интуиции и быстродействие машин. Кроме того, машина оформляет готовые чертежи и выдает их как рабочую документацию.

На быстродействующей печати печатаются техническая спецификация и все необходимые документы; проект готов.

В настоящее время ученые вынуждены сосредоточивать свои усилия на решении проектно-конструкторских задач в таких областях, где вопрос морального старения изготовляемого изделия стоит особенно остро. Например, электронно-вычислительная машина «МИР», творцы которой отмечены Государственной премией СССР за 1968 г., имеет размеры письменного стола, а техническая документация к ней составляет примерно три письменных стола, т. е. по весу и объему она больше, чем сама машина. Машина делается на микросхемах, а документация — по-прежнему. Чтобы такую документацию получить вручную, нужно около четырех лет

работы, а это срок морального старения вычислительной машины. Если не стать на путь автоматизации проектирования электронно-вычислительной машины, то возникает угроза, что машина в период окончания проектирования морально устареет. Это относится также к ряду других областей. Поэтому ученые концентрируют свое внимание, естественно, прежде всего на таких областях техники, где темп технического прогресса столь большой, сложность технических изделий столь велика, что без автоматизации проектирования нельзя успешно конкурировать с зарубежными странами в области технического прогресса. По мере того как будет создаваться кибернетическая индустрия, в нашей стране будет увеличиваться количество институтов, занимающихся разработкой таких проектов. Технологическая подготовка производства в настоящее время, к сожалению, автоматизирована несколько хуже. Институт кибернетики АН УССР выполняет научное руководство и основную часть работы по математическому обеспечению и алгоритмам при разработке систем управления для 16 крупнейших предприятий Советского Союза. В рамках этой работы, в частности, решается задача автоматизации технологической подготовки производства. Эта работа еще не закончена, первая очередь ее будет выполнена в конце 1970 г.

Коллектив ученых Института кибернетики решает задачи экономического управления, управления в истинном масштабе времени, оперативного управления в основном по двум направлениям. Это — управление различными технологическими процессами, т. е. создание систем, которые управляют отдельными объектами — домнами, фрезерными станками и т. д. Таких систем управления создано около двух десятков в разных городах, где они управляют процессами выплавки стали, вытяжки кинескопов, техническими процессами на содовом заводе и т. д.

Но сейчас главное внимание уделяется не столько управлению отдельными технологическими процессами, сколько управлению экономикой промышленности заводов в целом. Речь идет об управлении материально-техническим снабжением завода, об автоматизации бухгалтерского учета и т. д. Трудность создания таких систем заключается прежде всего в том, что при этом, как правило, существенно меняются существующие привычные методы управления. Дело в том, что без применения электронно-вычислительных машин человек каким-то образом приспосабливается к переработке в системе управления огромного потока информации за счет создания различных иерархических систем управления и соответственно распределения функций между ними.

Киевские кибернетики создают автоматизированные системы управления в основном в машиностроении, приборостроении и приобрели хороший опыт для решения различного рода задач в этой области. Все машиностроительные предприятия и предприятия приборостроения построены по такой схеме: есть сборочные цехи и различного рода вспомогательные и заготовительные цехи — механический, литейный, гальванический и т. д. На первый взгляд задача планирования и управления заводами кажется не очень сложной. Задается календарный выпуск каких-либо изделий. Если это массовое производство, то указывается, сколько соответствующих изделий нужно выпустить за квартал, месяц, какой заказ дается в механический цех, гальванический, литейный и т. д. А дальше необходимо решить, как делать изделия соответственно спецификации. При этом, как правило, применяется схема ручных методов управления, часто происходит несогласованность календарных планов и почасовых графиков работы отдельных производственных линий. Надо сказать, что наша

промышленность в масштабе всей страны от этой несогласованности несет колоссальные потери. По нашим подсчетам, если на крупнейших машиностроительных заводах страны ликвидировать только эту несогласованность в работе отдельных производственных линий, то темпы роста промышленности можно повысить вдвое при тех же пропорциях деления национального дохода, т. е. вместо 8—10 % прироста в год, которые мы имеем сейчас, можно получить 18—20 % прироста, не увеличивая доли накопления и оставляя долю потребления в национальном доходе на прежнем уровне.

Каково же происхождение этого резерва? Происходит все очень просто. Предположим, что литейный цех получил задание отлить 1200 деталей. Начальник литейного цеха проводит планирование либо с учетом степени готовности производства, либо с учетом финансовых соображений — обеспечения заработка соответствующим рабочим. В механическом цехе учитываются другие параметры, а в целом получается так, что деталь запланирована в литейном цехе на последнюю декаду, а в механическом цехе ее обработка запланирована в первой декаде месяца. Пока есть заделы — незавершенное производство предыдущего квартала,— механический цех работает ритмично. Но вдруг обнаруживается, что деталей нет, и начинается выяснение у директора, главного инженера, диспетчера причин срыва литейщиками производственного процесса. Такое положение существует между заводами, стройками и т. д. На заводе начинают предпринимать разного рода меры, переналаживать производственные линии, оборудование и т. д. со всеми вытекающими отсюда последствиями. Естественный способ борьбы с этим явлением — создание больших запасов незавершенного производства по каждой детали на каждой стадии обработки. За счет этих запасов цехи завода могут ритмично работать, но это явление имеет свою отрицательную сторону. Оказывается, что незавершенное производство у нас по стране превысило в целом все разумные нормы, какие когда-либо были известны. А это своеобразный тормоз на пути технического прогресса. Действительно, если завод из года в год выпускает одну и ту же продукцию и изменяет эту продукцию раз в десятилетие, тогда еще нет видимой опасности. Но современный машиностроительный или приборостроительный завод, особенно в передовых областях промышленности, имеет дело с изделиями, в которые конструкторские изменения вносятся десятками на протяжении одной недели. В каком положении оказывается дирекция завода? Очередное конструкторское изменение изделия позволяет делать его намного лучше, но на складах завода в запасе множество старых деталей, к примеру, на 10 млн р. Что делать в этом случае? Принять техническое новшество, отдать старый запас под пресс и нести финансовые потери или искусственно задержать внедрение нового технического прогресса, пока не иссякнут запасы, которые накопились? Значит, вопрос о создании запасов как буферов для устранения недостатков в материально-техническом снабжении как внутри завода, так и между заводами не является разумным путем решения проблемы. Более того, в настоящее время в мировой практике существует такая точка зрения, что только по некоторым видам изделий можно создавать материальные запасы. Рассмотрим, например, такую область техники, как электронно-вычислительные машины. Есть заводы, которые производят ферритовые кубы памяти. Сделать запас ферритовых кубов памяти, которые выпускают заводы для электронно-вычислительных машин, нельзя, так как это — быстро развивающаяся область.

Промышленные фирмы капиталистических стран, которые внедряют электронную технику, предпочитают в качестве резерва иметь не запасы

материалов соответствующих изделий, а запасы незадействованных производственных мощностей. Таким образом, управление запасами готовых изделий — это одна из очень сложных проблем, которая, как показывает исследование операций, в нашей промышленности решается еще неграмотно. Примеров грамотного решения этой проблемы мы практически не нашли ни на одном из обследованных заводов.

В чем состоит грамотность и неграмотность в решении этих задач?

Существуют разработанные математические методы, которые позволяют правильно определить размеры резервов. Один вид резервов необходим для покрытия недостатков в планировании, другой существует для непредвиденных случаев. Скажем, поломка станка ведет к прекращению выпуска на сборке, а это влияет на выпуск продукции всего завода и связанного с ним другого предприятия. Чтобы такие случаи устранить, органы управления — дирекция завода, министерство и т. д.— должны иметь соответствующие резервы, которые можно занести в действительные запасы, исходя из статистики ненадежности соответствующего оборудования.

Когда мы вплотную подходим к изучению нашей промышленности, например автомобильной и других, то видим, что по одним видам материалов у нас имеются запасы, которые в 100 раз превышают существующие научные нормы, а по другим — находимся на уровне 60 % необходимого. Но так как автомобили и тракторы нельзя выпускать даже без одной детали, то, несмотря на огромные запасы этих деталей на заводе, сборку лихорадит, станки стоят, останавливаются сборочные линии и т. д. Значит, правильное управление запасами резервов, оперативное приведение их в действие — это одна из важнейших задач. Как правило, оказывается, что для сколько-нибудь крупных производств с большим количеством связей такая задача без электронно-вычислительных машин не может быть решена.

Коллектив Института кибернетики АН УССР сейчас создает автоматизированные системы управления для 16 крупнейших заводов страны. Системы строятся таким образом, что обеспечивается автоматическое, не зависимое от воли и сознания людей, поступление текущей информации по состоянию дел на заводе в память электронно-вычислительной машины до самых мельчайших подробностей, т. е. электронно-вычислительная машина накапливает и хранит на своих магнитных лентах нормативы, план, трудовые ресурсы, состав оборудования, технические паспорта станков, прессов, состояние материальных запасов.

Как обеспечивается поступление этих данных в машину? Если в вычислительный центр доставлять сведения и после процесса перфорации вводить данные в машину, то такой способ будет несовершенен по двум причинам: во-первых, требуется определенное число работников, во-вторых, возникает дополнительный источник ошибок. Поэтому необходимо прежде всего обеспечить так называемый принцип одноразового ввода документов в машину и принцип совмещения ввода с изготовлением финансово-материальной отчетности. Делается это различными способами. Один из них такой, какой мы применили во Львове, когда вся постоянная информация закладывается в специальный телетайп, соединенный с электронно-вычислительной машиной. Машина принимает информацию и фиксирует ее.

Машина оперативно составляет план для каждой единицы оборудования, где рассчитано все до минуты. Причем она не забудет заблаговременно заказать все необходимое для изготовления детали в инструменталь-

ном, литейном цехе и т. д. В результате этого всякого рода несогласованности, которые забирают массу рабочего времени и вызывают простои оборудования, ликвидируются. Правда, надо отметить, что когда создается автоматизированная система управления в масштабе одного завода, то устраняются неполадки, которые возникают в результате плохого планирования на данном предприятии, но возникают новые, связанные с поставками. На Львовском телевизионном заводе, например, 400 поставщиков; от их регулярной работы зависит производственный ритм завода, и так на многих предприятиях.

Естественно, что задача автоматизации управления в полной мере может быть решена только в том случае, если создается система управления отраслью. Тогда на каждом предприятии или на объединении мелких предприятий существует вычислительный центр, в котором есть все сведения о работе этих предприятий. В соответствующем министерстве есть главный вычислительный центр, который связан с вычислительными центрами предприятий и регулирует их взаимную работу. В этом случае главной задачей министерства является не столько конкретное управление заводом, сколько урегулирование связей между заводами. Эта труднейшая задача, которая намного труднее, чем задача управления непосредственно заводом, пока не решена. Но в этом нет вины министерства или Госплана, так как объем информации, которую надо для этой цели переработать, такой, что ни один коллектив не в состоянии с ним справиться. Поэтому надо вооружить вычислительной техникой всю систему управления и планирования народного хозяйства.

Правда, во Львовской системе мы частично решили проблему материально-технического снабжения между заводами за счет того, что вводили в память машины телеграммы об отправке тех или иных грузов с номерами вагонов от поставщика, находящегося, например, в Сибири. Если он отправил контейнер с каким-то грузом, немедленно идет телеграмма во Львов и вводится в вычислительную машину. Машина примерно знает время движения поезда и ставит ожидаемую дату прибытия. Поэтому система может на месяц вперед дать любую справку директору, министру по поводу того, кто из поставщиков задерживает работу. Это, конечно, полумера, и в дальнейшем необходимо давать сведения о существующих ресурсах соответствующим директорам и министрам, чтобы в случае непредвиденных ситуаций этими ресурсами восполнить имеющийся недостаток материальных средств.

Еще одна большая трудность при автоматизации управления в промышленности состоит в том, что в машино- и приборостроении фактически в значительной мере потеряна технологическая специализация предприятия, т. е. план предприятий изменяется очень сильно в течение года, и структура плана плохо согласуется со структурой ресурсов.

Так, предприятие имеет определенное оборудование, например тяжелые фрезерные станки, станки токарные, сверлильные и т. д. В зависимости от того, какой план дан заводу — делать экскаваторы или тракторы,— и в зависимости от технологии соответствующего изделия в ход будут пущены те или иные ресурсы, в некоторое оборудование практически не будет использовано. Дирекции завода придется платить за простой оборудования, но она идет на это, так как нет гарантии в том, что через несколько дней министерство внесет изменения в план и фрезерные станки, которые простаивают, будут необходимы для изготовления продукции. Если сравнить политику наращивания ресурсов на предприятиях с политико

планирования продукции министерствами и Госпланом, то увидим, что в ней наблюдаются некоторые несоответствия: часто наращиваются те ресурсы, которые не соответствуют предполагаемой структуре технологического плана на последующие годы. В результате, хотя оборудование установлено, «узким местом» оказываются другие ресурсы, а это оборудование фактически простаивает. Так происходит и в случае планирования наилучшей загрузки оборудования с использованием электронно-вычислительной машины, поскольку отсутствует единая автоматизированная система управления экономикой на базе вычислительной техники. В соответствующих отраслях промышленности мы создаем такую систему слежения за использованием ресурсов различными предприятиями, политику систематического изменения этих ресурсов, чтобы распределение ресурсов и планирование необходимого оборудования делались в соответствии с долгосрочными планами технологической специализации предприятия. Это очень большой резерв в нашей промышленности, который может привести к увеличению производства.

Надо сказать, что опыт работы в области создания систем управления, который у нас уже есть, в настоящее время показывает, что чем больше масштабы соответствующей системы, тем больший экономический эффект от ее внедрения. Если каждый завод в стране будет снабжен автоматизированной системой управления, но между заводами связи будут управляться по-прежнему, то можно ликвидировать в лучшем случае 10 % тех потерь, которые сейчас несут предприятия, а 90 % останутся в силе. Поэтому главная задача состоит в том, чтобы создать глобальную систему управления.

В наших социалистических условиях очень важным вопросом является создание государственной системы управления. Отметим, что именно в создании общегосударственной системы управления сказывается преимущество социалистической системы ведения хозяйства. Например, в США широко используются вычислительные машины. Фирмы США внедряют электронно-вычислительные машины, создают гораздо более совершенные методы управления, чем европейские фирмы. Кстати, американцы получили возможность увеличивать на 50 % доходы фирм вследствие проникновения в Европу и организации новых совершенных форм управления. Но руководители фирм пытаются соединить системы управления между собой, и тут-то начинают проявляться недостатки капиталистической системы. Чтобы составить наилучший план развития одной фирмы, нужно знать планы конкурентов, а законы капитализма таковы, что эти планы конкуренты тщательно скрывают. В результате возникает новая отрасль промышленности — промышленный шпионаж, и электронно-вычислительные машины, установленные в фирмах, загружаются информацией не только о своих собственных планах, но и о планах конкурентов.

В нашей стране такого рода препятствия отсутствуют. Задача создания глобальной системы управления экономикой — одна из главных задач, которые стоят перед нашим обществом. При концентрации усилий и внимания к этой проблеме приблизительно данная задача может быть решена за 12—15 лет. Причем уже на первых этапах ее решения можно получить большой экономический эффект. В частности, данные экспертизы на Львовском телевизионном заводе показали, что вложение средств в систему управления оказалось в три раза более эффективным, чем вложение средств в основные фонды. Если бы руководство завода вложило эти денежные средства в строительство новых цехов, то оно получило бы эф-

фект в три раза меньший, чем при улучшении управления на данном пред
приятии.

Естественно, еще больший эффект будет получен на следующем эта
не создания автоматизированных систем. Ныне Академия наук Украин
празднует 50-летие. Ряд ученых Украины удостоены высокого звания Ге
роя Социалистического Труда. Институт кибернетики отмечен высше
правительственной наградой — орденом Ленина. Ученые-кибернетик
Советской Украины не пожалеют сил, здоровья, чтобы создать такую тех
ническую базу системы управления экономикой страны, которая буде
достойна нашего будущего коммунистического общества.

РАЗДЕЛ **2**

ТЕХНОЛОГИЧЕСКИЕ И ОРГАНИЗАЦИОННЫЕ АСПЕКТЫ СОЗДАНИЯ ЭВМ И ИХ МАТЕМАТИЧЕСКОГО ОБЕСПЕЧЕНИЯ

О НЕКОТОРЫХ ПРОБЛЕМАХ РАЗВИТИЯ ЭЛЕКТРОННОЙ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

(Вопросы кибернетики.— Вып. 20.— 1976)

На первом этапе развития электронной вычислительной техники и систем математического обеспечения преобладала тенденция к универсализации. Последний крупный шаг в этом направлении был сделан в ЭВМ третьего поколения (система ИБМ-360, ЕС ЭВМ и др.), в которых соединились два класса машин (для научных и для коммерческих применений), развивавшихся ранее отдельно.

Наряду с этим проявилась тенденция к специализации прежде всего в так называемых системах реального времени (управляющие ЭВМ, в частности бортовые). В настоящее время в связи с расширением областей применения ЭВМ и переходом к их преимущественному использованию в различного рода системах возникает острая необходимость в разумной специализации машинных комплексов и соответствующих систем математического обеспечения. Эта необходимость подкрепляется возможностью (возникшей в связи с успехами микроэлектроники) перекладывать все большую и большую часть задач, решавшихся ранее с помощью программ, на специализированные схемы и односторонюю (пассивную) память (с использованием принципа микропрограммирования). Решение подобной задачи затруднялось ранее малой надежностью и высокой стоимостью логических электронных цепей, не позволявшими создать надежные схемы с большим числом логических элементов.

Суть специализации, о которой идет речь, состоит, во-первых, в том, что для различных классов применений ЭВМ должны снабжаться различными системами периферийных устройств и, следовательно, специальными системами матобеспечения для управления взаимно согласованной работой этих устройств. Во-вторых, различные классы применений требуют различного подхода к организации комплексирования ЭВМ, ориентации отдельных процессоров на те или иные специальные функции (коммутация, первичная обработка информации, работа с графической информацией и т. п.). Это, в свою очередь, влияет на организацию систем данных, на системы команд и макрокоманд, организацию прерываний и, конечно, на структуру и функции операционных систем. Как показывает практика, разумная специализация операционных систем может в несколько раз повысить эффективность работы ЭВМ в определенных классах системных применений. Наконец, специализация технико-математических комплексов влияет на состав и организацию библиотеки стандартных программ. В число стандартных попадают многие достаточно специализированные программы. Что же касается управления данными, то в специализированных комплексах автоматизируются все процессы пополнения, обновления и контроля банков данных, а также процессы создания временных рабочих массивов, передачи данных от одних программ к другим и т. п.

Первоначально такие комплексы целесообразно создавать на базе обычных универсальных процессоров ЕС ЭВМ, комбинируя их, в случае необходимости, с микрокомпьютерами и управляющими ЭВМ. Специализация на этом этапе заключается в выборе состава периферийного оборудования и специализации операционных систем (прежде всего в части управления данными и ресурсами). Разумеется, должны быть решены вопросы стандартизации набора микрокомпьютеров и управляющих ЭВМ и их сопряжения с ЕС ЭВМ. В дальнейшем требования увеличения эффективности оборудования, а также упрощения программирования и облегчения общения с человеком приведут к специализации процессоров, хотя каждый из таких специализированных процессоров будет оставаться алгоритмически универсальным и потому в принципе пригодным и для других применений.

Уровень специализации технико-математических комплексов должен быть разумным в том смысле, что должен обеспечить возможность их эффективного использования в достаточно широком классе применений. Как известно, прогресс электронной вычислительной техники определяется, с одной стороны, возможностями технологии, а с другой — требованиями, вытекающими из практики применения ЭВМ. К сожалению, у нас до сих пор существует искусственный барьер между конструкторами ЭВМ общего назначения и требованиями, диктуемыми системными применениями ЭВМ. В результате при создании систем матобеспечения пользователи часто тратят напрасно силы, средства и время для решения программным путем тех вопросов, которые можно было бы легко и просто решить при соответствующем усовершенствовании схем. Для устранения этого барьера было бы очень целесообразно, чтобы созданием проблемно-ориентированных технико-математических комплексов занимались не пользователи, а конструкторы ЭВМ. Это, во-первых, позволило бы исполнять многие трудоемкие части будущих систем наиболее квалифицированными силами и одноразово. Во-вторых (и это, возможно, еще важнее), конструкторы ЭВМ получили бы новый мощный источник идей для совершенствования разрабатываемой ими техники. Без этого они будут вынуждены идти по пути копирования технических решений в тех странах и фирмах, в которых осуществляется более тесная и прямая связь разработчиков ЭВМ с практикой их применений.

Какие же типы проблемно-ориентированных технико-математических комплексов сейчас необходимо создавать?

В качестве первого из таких типов можно назвать комплексы, ориентированные на автоматизацию управления технологическими процессами автоматизацию сбора и обработки экспериментальных данных и испытаний сложных технических объектов. Среди вопросов, которые необходимо решать при создании таких комплексов, отметим, прежде всего, создание (на унифицированной элементной базе) ряда управляющих мини- и микрокомпьютеров и устройств связи с объектами, стандартизацию интерфейса с датчиками и исполнительными механизмами и создание соответствующей аппаратуры стандартизации форм представления аналоговой информации, поставляемой различного рода самопишущими регистрирующими приборами, и разработку комплекса быстродействующих устройств для ввода этой информации в ЭВМ. Необходимо также разработать эффективные операционные системы для иерархических многомашинных комплексов и принципы включения в системы аналоговых вычислительных устройств. Далее следует задача разработки пакетов программ для первичной обработки данных (сглаживание, интерполяция и т. п.). Потребуются

разработки устройств для убыстрения обработки информации в подобных комплексах. В качестве примеров можно назвать устройства для быстрого преобразования Фурье, параллельного опроса датчиков и т. п.

В отдельный класс можно выделить комплексы, предназначенные для подготовки данных и группового цифрового программного управления станками и другим оборудованием. Не говоря уже о специфических программно-языковых средствах, выделение этого класса в самостоятельный класс диктуется перспективами его дальнейшего развития. Как известно, эти перспективы определяются созданием и широким использованием промышленных роботов. Для роботов, снабжаемых «органами зрения» и системами тактильных датчиков, оказывается целесообразным создавать специальные схемы, облегчающие быстрое решение задач распознавания. Это же в какой-то мере относится к проблеме автоматической координации движений и создания набора соответствующих макрооператоров с учетом многих степеней свободы, которые имеют исполнительные органы современных роботов.

Комплексы, предназначенные для использования в системах организационного управления (АСУ), ставят перед конструкторами ряд новых задач. Помимо специфического набора периферийных устройств (регистраторов производства, специальных кассовых аппаратов, устройств для изготовления и чтения перфоярлыков и т. п.) здесь возникает задача создания специальных периферийных миникомпьютеров, ориентированных на первичную обработку больших массивов информации. Как и в предыдущих классах, важное значение имеет задача создания иерархически построенных многомашинных комплексов и соответствующих операционных систем. Задача управления данными не ограничилась автоматической идентификацией массивов данных и организацией их пересылок между внешней памятью и ОЗУ. Необходима гибкая система управления специальным пакетом программ для любых комбинаторных преобразований и любых агрегирований находящихся в комплексе массивов данных. Так как значительная часть задач в АСУ решается в рамках строгого расписания, операционная система должна иметь возможность готовить исходные данные к нужным моментам времени, вместо того чтобы начать их поиск и пересылку в ОЗУ в процессе работы программ по соответствующей макрокоманде (get в «OS-360»).

Должна быть разработана система ведения больших информационных массивов, включающая не только технические, но и организационные меры. Сюда относятся гораздо более развитые средства защиты информации от неправомерного обращения, чем средства, имеющиеся в «OS-360». Они должны допускать автоматическую проверку полномочий для доступа не просто к тем или иным массивам, а к результатам их обработки средствами системы (с соответствующим ограничением набора собственных средств пользователя).

Важное значение имеет организация обмена информацией между различными комплексами как через каналы связи, так и с помощью прямой пересылки машинных носителей. При этом необходимо использовать более удобные носители, чем ленты или пакеты дисков, например, начинающие сейчас завоевывать популярность дискеты. Отметим, что современные тенденции развития связи между ЭВМ укладываются в основном в две формы, представляющие аналоги обычной почтовой и телеграфной связи. Для ряда приложений в АСУ необходимо развивать аналог телефонной связи, при которой одна ЭВМ может обращаться к устройствам внешней памяти другой ЭВМ в обычном темпе их работы, допуская одновременное обраще-

ние второй ЭВМ к устройствам внешней памяти первой (режим дуплексной связи).

В связи с тем что главная перспектива развития АСУ связана с созданием Единой государственной сети вычислительных центров по обработке экономической информации, необходимо упомянуть еще хотя бы о двух задачах. Первая из них — это создание операционной системы такой сети, которая должна допускать создание любых временных конфигураций входящих в сеть проблемно-ориентированных комплексов и эффективное управление их совместной работой по решению сложных задач межведомственного характера. Такая работа не должна нарушать жизненно важных функций управления, выполняемых отдельными комплексами, и вместе с тем должна быть обеспеченной необходимыми ресурсами. Осуществить это можно с помощью специальной сети диспетчерских ЭВМ (с централизованным управлением и соответствующими средствами связи), которые из всех подведомственных им комплексов получали бы автоматически данные о планах работы, а также о текущем состоянии и загрузке имеющегося оборудования и вырабатывали бы оптимизированные планы управления создаваемыми конфигурациями.

Вторая задача — это управление распределенными банками данных. Она предусматривает создание специальных автоматических каталогов информационных массивов в информационно-диспетчерской части сети и иерархическое управление процессом поиска информации и формирования новых массивов из любых наборов территориально удаленных машинных массивов.

Особый класс проблемно-ориентированных комплексов представляют собой вычислительные центры коллективного пользования с доступом с удаленных терминалов. Помимо чисто технических вопросов (комплексирования мощных ЭВМ, связи абонентских пунктов и т. п.) здесь возникает ряд специфических задач по математическому обеспечению. Это, во-первых, создание такого управления ресурсами, которое обеспечивало бы высокую надежность работы комплекса, а, во-вторых,— разработка специальных языково-программных средств для ведения диалога с комплексом в режиме разделения времени с большого числа удаленных терминалов.

Важное значение имеют комплексы, ориентированные на справочно-информационные задачи большого объема. Здесь на первое место выдвигаются задачи создания устройств памяти большого объема (постоянные диски на сотни миллионов байт и др.), разумное сочетание машинной информации с системами микрофильмирования и библиотеками на микрофишах и т. п. Кроме того, возникает много вопросов, связанных с автоматическим считыванием документов, автоматическим индексированием, работой в естественных языках, вводом и выводом голосом и др. Важнейшее значение имеет такая форма организации данных и управления выборкой, которая обеспечивала бы как пакетную обработку поступающих запросов, так и быстрые ответы по отдельным срочным и особо срочным вопросам.

Очень большое значение имеют комплексы, ориентированные на автоматизацию проектно-конструкторских работ. Здесь имеются свои требования к периферийным устройствам и организации работы системы. Помимо обычных графопостроителей и графических дисплеев необходимы специальные конструкторские пульты с широкими возможностями работы с графической информацией (одновременный вывод трех проекций, возможность изменения освещения и точки обозрения общего вида представляемых на экране объектов, получение фотокопий с экрана и т. д.). Необ-

ходимы устройства для ввода графической информации, автоматического считывания координат и т. п.

Важнейшее значение имеет стандартизация формы представления графической информации в ЭВМ. Решение этой задачи позволяет создать пакет программ, позволяющий выполнять любые геометрические преобразования чертежей и рисунков (изменение масштабов, повороты, сдвиги, перевод из одной проекции в другую и т. п.), а также решать различного рода расчетные задачи, выбирая исходные данные для них непосредственно с чертежа.

Это могут быть как геометрические задачи (определение площади, объема, центра тяжести и т. п.), так и задачи теоретической механики, сопротивления материалов, электротехники и других общеинженерных курсов.

Для более специфических расчетных программ конструктор может пользоваться одним или несколькими проблемно-ориентированными языками и соответствующей системой автоматизации программирования.

Машинный архив чертежей стандартных деталей, блоков и устройств должен позволять конструктору осуществлять быстрый подбор нужных элементов и собирать из них требуемую конструкцию, осуществляя их перенос на нужное место экрана с помощью светового карандаша. Среди этих элементов могут быть и чисто геометрические объекты (дуги окружности, отрезок прямой и т. п.).

Поскольку современные ЭВМ плохо приспособлены для представления графической информации, работа по созданию комплексов для автоматизации проектно-конструкторских работ должна оказать наибольшее влияние на дальнейший прогресс архитектуры и системы математического обеспечения ЭВМ.

Все описанные комплексы будут иметь весьма широкий класс применений. В качестве примера более специального комплекса можно привести комплекс, ориентированный на задачи автоматизации перевода и редактирования. Его основой является комплекс ЭВМ со специальными пультами редакторов-переводчиков. На входе системы должны работать читающие автоматы, настраиваемые на различные шрифты. После прочтения первой страницы машинописного или печатного текста универсальные ЭВМ комплекса анализируют шрифт и настраивают на него входной читающий автомат. Далее идет быстрое считывание всего текста.

На пульте редактора на специальном дисплее отображаются одна за одной страницы или абзацы редактируемого текста. Редактор с помощью светового карандаша и клавиатуры может осуществлять различные редакторские операции (вставить или изъять слово, переставить слова, изменить или вставить тот или иной знак и т. п.). Отредактированные части текста направляются на управляемую ЭВМ наборную машину. В случае перевода с иностранного языка на пульте редактора-переводчика отображается как исходный иностранный текст, так и его перевод, выполненный (не обязательно качественно) каким-либо машинным алгоритмом.

Разумеется, работа по проблемно-ориентированным комплексам, несмотря на всю ее важность, не должна заслонять от нас актуальные задачи развития вычислительной техники как таковой, а также работы теоретического характера, определяющие пути совершенствования ЭВМ и систем общего математического обеспечения.

Проблема надежности ЭВМ по-прежнему продолжает оставаться весьма актуальной. Помимо обычных методов ее решения (повышения надежности элементов дублирования, резервирования, аппаратного и програм-

много контроля) в последнее время развивается ряд новых методов. Это прежде всего использование во всех цепях машины кодов с исправлением ошибок, что приводит к гораздо более экономным решениям, чем, скажем, утроение аппаратуры, используемое в особо ответственных управляющих ЭВМ. Второе направление — использование блочного резервирования с самонастройкой (изменением функций блоков). При этом возникновение отказов приводит к уменьшению производительности ЭВМ, но не за счет важнейших из выполняемых ею функций, для которых уровень резервирования является гораздо более высоким, чем для более второстепенных функций.

Использование указанных двух методов позволяет создавать ЭВМ со временем безотказной работы по основным функциям, исчисляемым десятками лет.

Задача повышения быстродействия ЭВМ за счет скорости работы ее элементов требует, помимо традиционных, использования новых физических эффектов. Весьма перспективным направлением является, по-видимому, использование эффекта Джозефсона, одного из наиболее быстродействующих физических процессов, который можно использовать как в запоминающих устройствах, так и в логических цепях. В связи с тем что сегодня уже решены многие принципиальные вопросы, позволяющие строить оптические вычислители, представляет интерес повышение частоты работы оптических элементов (имеется в виду, разумеется, частота следования дискретных импульсов света, а не его несущая частота). Огромное значение имеет нахождение таких физических эффектов, в которых в результате взаимодействия полей осуществлялась бы параллельная обработка дискретной информации. Пусть, например, импульс света, несущий информацию о матрице A, состоящей из нулей и единиц, под воздействием другого (управляющего) импульса света осуществляет определенное преобразование $A \to f(A)$ этой матрицы. Если бы в нашем распоряжении был набор управляющих импульсов, способных осуществить полную систему таких преобразований, то мы могли бы построить оптическую матрицу, фактическое быстродействие которой было бы намного выше, чем частота следования импульсов.

По мере новых успехов в микроэлектронике (прежде всего в технологии производства больших интегральных схем) все ближе к непосредственному практическому использованию становятся идеи, развиваемые в теории вычислительных сред.

Создание недорогой оперативной памяти большого объема (порядка 10^{10}—10^{13} байт) явилось бы настоящей революцией в вычислительной технике, полностью изменив нынешние взгляды на архитектуру и операционные системы ЭВМ. Большие перспективы здесь открываются за счет использования голографии (с быстродействующими обратимыми пленками) и уже упоминавшегося эффекта Джозефсона.

Магнитодоменная память призвана заменить громоздкую и неудобную внешнюю память на магнитных лентах и магнитных дисках. Хорошо согласуясь (без применения специальных буферов) с любыми скоростями обращения, такая память является идеальной для различного рода автономных устройств подготовки данных и обмена информацией (путем транспортировки носителей) между ЭВМ.

По-прежнему одним из важнейших направлений совершенствования ЭВМ является упрощение общения человека с машиной. Успехи технологии БИСов и предоставляемые ею возможности дальнейшего усложнения схем позволяют по-новому подойти к решению таких проблем, как

проблемы ввода в ЭВМ информации голосом, распознавания рисунков, оперирования трехмерными цветными изображениями, работы с машиной на естественных человеческих языках и т. п. Одной из ближайших задач является усовершенствование систем «диалог», «человек — машина» по линии как технических средств и организации системы, так и языков диалога.

Повышение «интеллекта» машин в наши дни происходит в основном за счет приближения внутренних машинных языков к проблемно-ориентированным языкам высокого уровня. Это направление, зародившееся в нашей стране (в серии машин «Промінь» — «МИР»), в настоящее время (после его признания на Западе) получило у нас широкое распространение. Дальнейший прогресс в этом направлении мы связываем прежде всего с развитием таких направлений применения ЭВМ, как автоматизация доказательства теорем и других логических построений.

Как известно, развитие этого направления до сих пор шло по пути поиска универсальных доказывающих процедур для классической математической логики. Как и следовало ожидать, на этом пути хорошие результаты были получены лишь в рамках самой математической логики, раскладывающей доказательства на мельчайшие элементарные кирпичики. В содержательных разделах математики, где используются гораздо более крупные строительные блоки, нужен совершенно другой подход. В Институте кибернетики АН УССР была построена практическая математическая логика, которая относится к классической математической логике примерно так же, как современный язык программирования высокого уровня (например, АЛГОЛ или ПЛ-1) относится к языку простейшей машины Тьюринга.

Помня о роли, которую сыграла классическая математическая логика в становлении современных ЭВМ, нетрудно понять, что построенная практическая математическая логика поможет установить, в каком направлении следует развивать архитектуру ЭВМ, чтобы они в полной мере заслуживали название искусственного мозга.

Разумеется, при проектировании подобных машин с высоким уровнем интеллекта резко возрастает объем работ по проектированию на логико-алгоритмическом уровне. Особенно серьезным является тот факт, что при таком проектировании обычная интуиция, достаточно хорошо работавшая для таких простейших схем, как сдвиговые регистры, сумматоры и т. п., уже не приводит к сколько-нибудь удовлетворительным решениям. Поэтому необходимо иметь такую систему автоматизации проектирования ЭВМ, которая позволяла бы осуществлять формальные преобразования и оптимизацию схем и алгоритмов без изменения выполняемых ими содержательных функций. Такая система разработана в Институте кибернетики АН УССР и успешно работает. На очереди — интеграция этой системы с системами технического проектирования (имеющимися сегодня во многих институтах и КБ) и с системой автоматического изготовления сложных микроэлектронных схем.

(Управляющие системы и машины.— 1977.— № 5)

Новый огромный скачок в экономике, который предстоит совершить нашей стране в десятой пятилетке, требует дальнейшего ускорения темпов научно-технического прогресса как решающего условия повышения эффективности общественного производства и улучшения качества продукции. Последовательно решать задачу органического соединения достижений научно-технической революции с преимуществами социалистической системы хозяйства — такова директива Коммунистической партии.

Одним из главных инструментов современного научно-технического прогресса служит электронная вычислительная техника. Решениями XXV съезда КПСС в текущем пятилетии предусмотрено увеличить выпуск средств вычислительной техники в 1,8 раза. Поставлены большие задачи по дальнейшему применению электронно-вычислительных машин (ЭВМ) в научных исследованиях, на производстве, в экономике.

Наша страна имеет значительный опыт решения сложных научных, проектно-конструкторских, планово-экономических и других задач на ЭВМ. Однако этот опыт накоплен в основном на ЭВМ второго поколения («БЭСМ-6», «М-220», «Минск-32» и др.), а его уже недостаточно, поскольку, как известно, в минувшей пятилетке наша промышленность перешла на выпуск единой системы электронно-вычислительных машин третьего поколения, машинный язык которой (т. е. система выполняемых ею элементарных операций) существенно отличается от языка всех предшествующих машин. Важно и то, что переход к единой системе ЭВМ коренным образом меняет условия использования вычислительной техники, включая технику программирования, обработку данных, организацию вычислительного процесса, характер взаимодействия потребителя с ЭВМ и т. д. Все это обусловлено тем, что электронно-вычислительная техника из «полуэкзотических» научных инструментов для решения особо сложных задач перешла в разряд средств, обслуживающих массовых (притом весьма разнообразных) потребителей в режиме поточного производства.

Мировой опыт показывает, что этот переход явился мощным фактором нового роста темпов научно-технического прогресса. Он не ограничивается рамками научно-исследовательских и проектно-конструкторских организаций, а охватывает ныне автоматизацию промышленного производства, управление экономикой и другие сферы общественной практики. Научно-техническая революция обусловила необходимость создания новой отрасли индустрии — переработки информации. Начало формированию ее в нашей стране было положено в предыдущих пятилетках (главным образом в девятой). Это сложный процесс, сопряженный с большими трудностями.

Назовем основные из них.

Продолжает остро ощущаться недостаток электронно-вычислительных машин ряда важных классов, прежде всего ЭВМ большой мощности, а также дешевых, малогабаритных мини- и микроЭВМ. Недопустимо задержалось решение вопросов унификации последних и четкой специализации министерств (радиопромышленности, приборостроения, средств автоматизации и систем управления, электронной промышленности и др.) в деле разработки и производства мини- и макроЭВМ различных классов. Отсюда — неоправданное удорожание всего жизненного цикла ЭВМ (разработка — производство — эксплуатация). Медленно решаются вопросы соединения электронно-вычислительных машин различных классов в системы (комплексы) для совместной работы, достижения таких возможностей, которыми эти машины порознь не обладают. Особенно это касается комплексирования больших универсальных систем ЭВМ со специализированными мини- и микроЭВМ. В большинстве случаев подобные комплексы создаются самими потребителями, что распыляет усилия, приводит к значительному удорожанию разработок и затягиванию сроков их исполнения.

Как и всякое поточное производство, современный вычислительный центр представляет собой сложную систему, состоящую из множества различных узлов (от центральных устройств, реализующих собственно процесс переработки информации, и электронной памяти от автоматических устройств для резки бумаги, специальных стеллажей для хранения магнитных лент, дисков, перфолент, перфокарт, тележек для их перевозки и т. д.). Как и на всяком производстве, количество этих устройств и их пропускные способности должны быть строго сбалансированы. Одно «узкое место», один плохо автоматизированный или механизированный участок могут резко снизить эффективность работы всей системы.

Дефицитным является общесистемное периферийное оборудование (необходимое для вычислительных центров всех классов). Неважно обстоит дело и с производством специального периферийного оборудования вычислительных центров, предназначенных для комплексной автоматизации управления сбором и обработкой массовых экспериментальных данных, для автоматизации проектно-конструкторских работ и т. д. В то же время создавать сегодня вычислительные центры, не оснащая их полным комплексом современного периферийного оборудования, все равно, что строить заводы, скажем, без необходимой подъемно-транспортной техники.

Для индустрии и переработки информации положение усугубляется тем, что управление ею намного сложнее управления любой известной сферой материального производства. Это связано не только со сложностью самих процессов информационного производства, но и с огромной быстротой их протекания и изменения. Поэтому все оборудование вычислительных центров останется мертвой грудой металла до тех пор, пока не будет разработана и приведена в действие сложная система программ (так называемая операционная система), которая управляет всеми процессами, реализуемыми на этом оборудовании, организует поиск данных и обмен данными между устройствами, диалог системы с человеком и т. д.

Операционная система относится к так называемому **внутреннему математическому обеспечению** ЭВМ или комплексов ЭВМ. К нему сегодня принято относить также системы автоматизации программирования. Смысл этих систем заключается в том, чтобы конкретные программы обработки данных можно было представить в форме, удобной для тех, кто обращается к машине, а затем автоматически преобразовать в форму, понятную для

ЭВМ, но крайне не удобную для человека. Без таких конкретных программ и исходных данных к ним вычислительный центр уподобляется заводу, на котором смонтировано все оборудование и налажено управление, но которому не задано самое главное: какую продукцию и в какое время он должен выпускать (нет ни рабочих чертежей изготовляемых изделий, ни календарного плана их производства), а также не указаны сроки, когда и какое материально-техническое снабжение (при наличии вычислительных центров — исходные данные) он должен получать.

Степень автоматизации программирования (а следовательно, и простота составления программ для тех, кто пользуется ЭВМ) зависит от уровня специализации вычислительного центра. Ясно, что если здесь решается строго ограниченный круг задач, то их программы должны быть написаны заранее — раз и навсегда — и введены в память ЭВМ (сформировав так называемую «библиотеку программ»). В этом случае достаточно указать наименование требуемой программы и ввести в машину необходимые данные.

Нередко (например, в автоматизированных системах управления) удается частично или полностью автоматизировать и процесс подготовки исходных данных. Поступая в систему от различного рода источников (автоматических датчиков, других вычислительных центров или, наконец, от людей), данные автоматически формируются в так называемый банк данных. В банке вводится система условных обозначений (имен) для групп данных, необходимых при решении тех или иных задач. Для решения задач в специализированном вычислительном центре с банком данных достаточно сообщить операционной системе имя рабочей программы (или последовательности таких программ), имена необходимых групп данных из банка (а также, возможно, некоторые дополнительные данные) и указать желательную форму выдачи результатов.

Процесс программирования для тех, кто непосредственно пользуется ЭВМ, здесь почти полностью упразднен, разумеется, вследствие огромной предварительной работы программистов. Выгодность такого подхода очевидна: программы, созданные один раз высококвалифицированными программистами (и потому, как правило, высококачественные и эффективные), могут многократно эксплуатироваться работниками, практически не имеющими никакой специальной подготовки. В ряде случаев, например при автоматизации многих технологических процессов, удается полностью исключить вмешательство человека в работу системы: исходные данные, поступая в систему, сами включают программы для своей обработки.

Еще один крайний случай — вычислительный центр общего пользования с широким (заранее не прогнозируемым) спектром решаемых задач. Однако даже здесь оказывается возможным создать и эффективно использовать библиотеку программ, которые часто применяются либо сами по себе, либо в качестве модулей в более сложных программах. Такие программы объединяются в соответствии с их целевой направленностью в так называемые пакеты. Фактически это предварительные программные заготовки, которые системой автоматизации программирования могут быть превращены в рабочие (машинные) программы.

Использование предварительных программных заготовок, поставка их вместе с ЭВМ и централизация процесса последующего программирования и обмена программами — все это своеобразная форма унификации и стандартизации в среде информационной индустрии. Эта задача неразрывно связана с унификацией и стандартизацией форм представления данных не только на входе и выходе, но и внутри ЭВМ. Поэтому, например, пакет

программ, созданный применительно к одним формам планово-учетных документов, может оказаться неприемлемым при других формах документов. Кроме того, формы документов, рассчитанные на людей, часто оказываются плохо приспособленными для ЭВМ, неоправданно усложняя работу с ними. Решение же задач унификации и тем более изменения форм документов упирается в многочисленные межведомственные препоны, особенно если принять во внимание, что это не входит в функции министерств, производящих вычислительную технику.

Одно из важнейших условий преодоления перечисленных трудностей — решительный отказ министерств (производителей вычислительной техники) от укоренившейся тенденции поставлять потребителям отдельные ЭВМ (с минимальным математическим обеспечением). Необходимо как можно быстрее переходить к практике разработки, поставки, сборки и наладки у потребителей полных комплексов технических и программных средств, составляющих законченные автоматизированные центры общего и специального назначения. К этому нужно добавить также централизацию служб технического обслуживания, ремонта и модернизации созданных систем, организацию обучения технического персонала потребителей с целью наиболее эффективного использования и т. д.

Хотя известные шаги в направлении решения некоторых из этих задач уже сделаны, в целом проблема перехода к политике комплексных разработок и поставок автоматизированных систем обработки данных еще далека от решения. Сегодня большинство потребителей создает такие системы не сразу на основе полного проекта, с помощью одной головной подрядной организации, а в результате мучительного процесса «проб и ошибок», взаимодействуя последовательно с десятком (а то и с несколькими десятками) независимых поставщиков и подрядчиков. Приобретая ЭВМ в минимальной комплектации и с минимальным математическим обеспечением, потребитель обычно убеждается в том, что он не может с ее помощью успешно решать стоящие перед ним задачи; кроме того, у него чаще всего (особенно, когда ЭВМ новейшей конструкции) нет своих кадров, способных эффективно использовать это оборудование. И вот начинается долгий и нелегкий процесс доработки системы, оснащения ее всем необходимым, обучения кадров, создания недостающих программ. Поскольку же обязанности по производству и поставке многих видов оборудования и программ для систем обработки данных между ведомствами, предприятиями и институтами строго не распределены, потребитель вынужден размещать свои заказы, где придется, и мириться с тем, что его система остается не вполне укомплектованной, а следовательно, и не вполне эффективной.

Ясно, что подобными методами нельзя создать полноценную индустрию переработки информации. В лучшем случае они пригодны для организации кустарных мастерских, а не современного поточного производства. Комплексная разработка и поставка законченных систем обработки данных (особенно в случае, когда ответственность за это будет возложена на одно специализированное министерство) — вот что кардинальным образом может улучшить положение с разработкой и внедрением новейших средств вычислительной техники.

Тут важно подчеркнуть еще одно обстоятельство. Отвечая за конечный результат, разработчики и производители ЭВМ были бы вынуждены устранять недостатки своей техники, выявившиеся при работе в реальных системах, и соответствующим образом ее совершенствовать.

С аналогичными трудностями встречаемся и при производстве программ. У нас нет даже полного учета их, не говоря уже о разумной специа-

лизации и кооперации коллективов, способных разрабатывать современное математическое обеспечение ЭВМ и их систем. Кроме того, далеко не все эти коллективы (особенно вузы) своевременно получают ЭВМ новейших конструкций, хотя именно они в первую очередь нуждаются в хорошем техническом обеспечении. По-видимому, пора создавать региональные центры разработки математического обеспечения новых ЭВМ, закрепив за ними соответствующие коллективы Академии наук СССР, Министерства высшего и специального образования СССР и отраслевых министерств. Первые образцы ЭВМ необходимо устанавливать в этих центрах, а после накопления достаточного объема математического обеспечения поставлять данные машины потребителям.

При должной организации работы отрасли, производящей ЭВМ и автоматизированные системы обработки данных (программно-технические комплексы), рассчитанные на разные классы применений, можно эффективно решать разнообразные проблемы внедрения вычислительной техники. Однако простая сумма пусть даже значительного числа участков, цехов и предприятий по автоматизированной обработке информации еще далеко не является той новой отраслью (информационной индустрией), о которой мы здесь говорим. При объединении всех этих ячеек в единую систему рождается новое качество и неизмеримо возрастает эффект автоматизации обработки информации.

Мировая практика показывает, какие огромные возможности открываются в случае, когда отдельные очаги автоматизации обработки информации сливаются в сети, работающие (через системы автоматической связи) под единым управлением. Однако у такой сети (точнее, у ее диспетчерской службы) должен быть один хозяин. То же относится к мощным территориальным вычислительным центрам коллективного пользования, обслуживающим по каналам связи отдаленных потребителей, не имеющих собственной мощной вычислительной техники.

Наконец, планирование очередности автоматизации по отраслям производится сегодня без глубокого учета существующих между ними взаимосвязей. В результате часто работа автоматизированного цеха и даже целого предприятия, в которых производственные операции рассчитываются по минутам и даже секундам, срывается по вине поставщика, который по старинке обходится месячными и даже квартальными планами производства и поставок продукции. Естественно, что окончательно такие вопросы должен решать Госплан СССР. Однако он не может этого сделать без мощной научной базы. Применительно к планированию развития других отраслей Госплан решает подобные проблемы через сами отрасли с помощью их институтов, информационная же отрасль организационно не оформлена и своих институтов не имеет.

Из сказанного напрашивается вывод, что информационную отрасль необходимо организационно оформить — быть может, в качестве специального ведомства со своими специфическими функциями, иными, нежели у отраслей, производящих ЭВМ и автоматизированные системы обработки данных. По отношению к ним эта отрасль должна выступать в роли генерального заказчика, не только направляющего и объединяющего заказы других отраслей, но и формирующего свой собственный заказ на объекты, поступающие непосредственно под его юрисдикцию (автоматизированная общегосударственная информационно-диспетчерская служба, сеть вычислительных центров коллективного пользования и т. д.). Тем самым была бы создана надежная основа для успешного выполнения поставленной партией задачи: обеспечить дальнейшее развитие и повышение эф-

фективности автоматизированных систем управления и вычислительных центров, последовательно объединяя их в единую общегосударственную систему сбора и обработки информации для учета, планирования и управления. Органическое единство развития технической базы автоматизированной обработки информации с дальнейшим совершенствованием экономических механизмов и организационных структур управления существенно повысит эффективность АСУ и всей информационной отрасли в целом. Обеспечение такого единства — важнейшая задача.

В функции информационной отрасли должны входить поиск новых областей эффективного применения ЭВМ и оказание помощи тем отраслям, которые не имеют достаточного научного потенциала для такого поиска.

У производителей ЭВМ и систем тоже немало забот. В десятой пятилетке необходимо осуществить переход на новую техническую базу — ЭВМ четвертого поколения на больших интегральных схемах. Технология таких схем позволяет решить две важнейшие задачи: во-первых, создание супермашин, производительность которых составит многие десятки миллионов операций в секунду; во-вторых, массовое производство дешевых и высоконадежных микрокомпьютеров для широкого применения. Последние благодаря миниатюрным размерам и дешевизне открывают совершенно новые пути в автоматизации.

Отметим, что создание и производство ЭВМ и систем четвертого поколения невозможны без комплексной автоматизации как процесса их проектирования, так и изготовления больших интегральных схем, монтажа и контроля их на всех этапах технологического процесса. В этом направлении советскими учеными обеспечен большой научный задел, который уже в значительной мере воплощен в действующие системы.

В десятой пятилетке предстоит выполнить значительный объем научных работ, который явится основой создания ЭВМ пятого и последующих поколений и откроет новые возможности для их эффективного применения.

НЕКОТОРЫЕ ЗАДАЧИ СОЗДАНИЯ И РАЗВИТИЯ МЕТОДОВ И СРЕДСТВ КИБЕРНЕТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ, СТОЯЩИЕ ПЕРЕД МОЛОДЫМИ УЧЕНЫМИ ИНСТИТУТА КИБЕРНЕТИКИ АН УССР

(Кибернетика.— 1978.— № 5)

Современный этап развития кибернетической науки характеризуется, пожалуй, самым интенсивным использованием средств вычислительной техники. При этом существенно, что в одном эксперименте, одном исследовании приходится использовать большое разнообразие средств, которые чаще всего создаются в процессе работы. Это приводит к тому, что современным исследователям-кибернетикам необходимо владеть искусством системного программирования. Такой подход предполагает наличие у исследователя или экспериментатора не только навыков работы с одним видом вычислительных систем или некоторым набором средств автоматизации программирования и организации вычислительного процесса в ЭВМ, но и глубокого знания и понимания внутренних механизмов работы современных вычислительных систем и способов эффективной их настройки или даже перестройки под данный класс задач.

В истории программирования было несколько периодов. Каждый из них характеризуется своим набором средств программирования. Развитие этих средств идет, с одной стороны, в направлении приближения их к понятиям, с которыми встречается пользователь, а с другой — требования по эффективности решения задач на ЭВМ приводят к тому, что в этих средствах находят отражение свойства и особенности структуры и внутренних (машинных) механизмов организации вычислений. Таким образом, современному программисту, какую бы программу он ни разрабатывал (будь это задача идентификации объектов в АСУ, распознавания образов, решения системы уравнений в линейной алгебре или оптимизации на графах), приходится, кроме пакета программы, собственно реализующих алгоритм решения соответствующей задачи, создавать еще специальную управляющую программу (а иногда специализированную операционную систему), диалоговую систему общения пользователя с программами разрабатываемого пакета и другой необходимый сервис. Я уже не говорю о том, что для класса задач почти всегда целесообразно иметь специальную систему программирования, использование которой существенно повышает производительность труда программиста.

По квалификации программистов можно разделить на три категории: системные программисты, работающие в области совершенствования технологических средств программирования в современном смысле этого слова, программисты, которые владеют некоторой технологией и могут оформлять результаты деятельности в той или иной конкретной области в виде пакетов или специальных систем программ, и, наконец, пользователи широкого плана, обладающие минимумом знаний о программировании, но являющиеся специалистами в конкретной области. Отмечу, что для современного исследователя-кибернетика, ведущего экспериментальные работы на ЭВМ, программирование является необходимым инструментом и

чем выше квалификация программиста, тем быстрее он может прийти к цели своих исследований.

В нашем институте выполняется ряд работ в области создания новых технологических средств программирования. Разработан метод формализованных технических заданий проектирования программ, Государственной комиссии сдан КТК-технологический комплекс программиста, развивается автоматизированная система производства программ и др.

Вместе с тем ведутся работы по созданию и освоению банков данных, в частности, по системам «Ока», «Пальма» и т. д.

Очень важной задачей является обеспечение таких условий работы программистов, при которых эти системы были бы «завязаны» в единый комплекс.

Наш вычислительный центр в настоящее время переживает этап, когда формируются новые условия работы и новые требования к пользователям. Это связано с большой исследовательской и экспериментальной работой, которую он проводит по освоению новых средств ЕС ЭВМ, разрабатываемых в рамках СЭВ (соответствующее соглашение по этому поводу было принято недавно).

Кроме того, в вычислительном центре намечается резкое увеличение оснащенности периферийным оборудованием, прежде всего дисками большой емкости. В связи со сказанным одной из важнейших является задача комплексирования оборудования вычислительных машин. При этом важно не только решение данной задачи применительно к штатному оборудованию ЕС ЭВМ и оборудованию разнородных машин с системами команд, отличными от ЕС ЭВМ, но также продолжение и реализация идеи комплексирования применительно к программному оборудованию. Эта же задача стоит и в связи с дальнейшим превращением нашего ВЦ в высокопроизводительный центр коллективного пользования с удаленным доступом. Участие молодых исследователей в такой большой и важной работе было бы весьма желательным. Отметим, что эти работы связаны с определенными неудобствами и трудностями — почти непрерывной сменой машин и практической несовместимостью программного оборудования при переходе от нижних моделей ЕС ЭВМ к верхним (чего, в принципе, не должно быть).

Вопросам автоматизации научных исследований в настоящее время уделяется большое внимание. Именно этот вопрос обсуждался на первой встрече президентов академий наук стран — членов Совета экономической взаимопомощи, которая не так давно состоялась в Москве. Решение этой проблемы существенно зависит от правильного использования вычислительной техники. Она включает в себя не только и даже не столько решение задачи обработки результатов экспериментальных исследований, но и целый ряд других задач исследовательского и особенно организационного характера, таких, например, как создание специальных программных систем и пакетов, о которых я уже упоминал, упорядочение расчетных работ, создание мощных вычислительных центров коллективного пользования, создание сетей ЭВМ, включающих специализированные банки данных, ориентированные на определенную область исследований, и др. В частности, в Академии наук УССР начато создание нескольких таких банков, например, в Институте проблем материаловедения. Перед нами стоит задача разработки и внедрения соответствующего программно-технического сервиса для таких работ, прежде всего для Академии наук УССР. Вопросы создания сети ВЦ интересны и тем, что в настоящее время имеются предпосылки для подключения ее к европейским сетям, а через них —

к американским системам. Эта перспектива интересна, с одной стороны, возможностью использования новых вычислительных мощностей, а с другой — объединением усилий социалистических стран в создании специализированных банков данных. Кроме того, это связано с использованием международных стандартов в информатике, а также с обменом и продажей программно-технического оборудования ЭВМ.

Рассмотрим задачи, появляющиеся в связи с разработкой автоматизированных систем управления различных классов. Основное внимание здесь переносится на решение задач по созданию программно-технических комплексов, нацеленных на классы применений. Если говорить об АСУП, то приходится думать о создании даже интегрированных программно-технических комплексов, в которых, кроме задач управления, решаются задачи проектирования и задачи автоматизации испытаний изделий в комплексе. Подступы к решению этой сложной задачи еще предстоит разведать. Напомню уровни интеграции в системах управления: первый — интеграция информационной базы; второй — интеграция организационного управления с управлением технологическими процессами; третий — добавление к уже имеющимся этапам проектирования и испытаний изделий. Для осуществления первого уровня в настоящее время для ЕС ЭВМ имеются инструментальные средства, хотя еще недостаточно освоенные и совершенные. По второму и третьему уровням еще многое предстоит сделать. Для осуществления этого необходимо уметь решать задачи создания программно-технических комплексов. Здесь возникает целый ряд сложных вопросов, среди которых отмечу следующие: комплексирование программно-технического оборудования вычислительных средств, генерирование операционных систем для таких комплексов, решение задач унификации технического оборудования и, что особенно важно, создание соответствующих средств стыковки. Аналогичные задачи появляются при создании систем автоматизации проектирования различных классов. В нашем институте ведутся работы по системам автоматизации проектирования. Я имею в виду систему ПРОЕКТ (автоматизации проектирования вычислительных машин), автоматизированную систему проектирования строительных конструкций, систему автоматизации проектирования в машиностроении и др. Развитие этих систем связано, во-первых, с переходом на новые машины и, во-вторых,— с расширением их возможностей в связи с работами по АРМам и другому периферийному оборудованию. Задача унификации при проведении всей этой работы, а также при создании программно-технических комплексов, позволяющих быстро оснастить любое конструкторское бюро, независимо от его размеров и профиля (в рамках машиностроения, приборостроения и строительства), чрезвычайно важна. В связи с этим предстоит выполнить очень большой объем работ как по созданию соответствующих средств, так и по конкретной реализации программы научных экспериментов. Очень важной задачей в рамках создания программно-технических комплексов является развитие системы БАРС. Напомню, что этот программируемый мультиплексор чрезвычайно важен для решения задачи автоматизации документооборота. БАРС — универсальное средство программируемой стыковки системы вводных устройств разных классов. В настоящее время одна из задач состоит в том, чтобы осуществить реализацию этой системы на микропроцессорах.

В работе по миниЭВМ можно выделить следующие задачи:

1) освоение СМ и миниЭВМ, уже выпущенных или выпускаемых промышленностью в качестве составных элементов вычислительных систем;

2) определение и характеристики применений мини- и микроЭВМ;

3) разработка новых мини- и микроЭВМ, в некотором смысле продолжающих линию машин серии «МИР». При этом желательно, чтобы эти разработки были использованы в работах по СМ ЭВМ. Сюда же относятся работы по коллективным интеллектуальным терминалам, которые должны существенно облегчить общение пользователя с вычислительной машиной.

В рамках работ по рекурсивным ЭВМ продолжает оставаться актуальной задача по распараллеливанию управления. Она тесно связана с перспективой создания новых типов микропроцессоров и БИСов. Важной является задача коммутации большого числа процессорных элементов. Интересно также направление работ по автоматизации проектирования многопроцессорных систем и, следовательно, по созданию соответствующих средств автоматизации.

Рассмотрим теперь задачи, возникающие в направлении исследований по искусственному интеллекту. В настоящее время в нашем институте выполняется ряд пока независимых исследований, хотя впоследствии планируется постепенное их сведение в единый комплекс. Среди них одной из важнейших является работа по автоматизации поиска доказательств теорем и обработке математических текстов. Трудоемкой и в то же время интересной работой является фактическое заполнение информационных массивов математическими текстами и совершенствование системы диалога математика-исследователя с разрабатываемой на ЕС ЭВМ программной системой. Большое значение имеет становление исследований по созданию машинных моделей «мира». Имеются в виду, конечно, ограниченные «миры» и средства их описания. Цель этих исследований состоит в использовании их результатов для дальнейшего продвижения вперед в таких областях искусственного интеллекта, как роботы, распознавание образов, автоматизация перевода с одного языка на другой, принятие решений в сложных ситуациях и т. п. Причем поскольку модель описывает ограниченный «мир», и, следовательно, семантические связи уже как бы заказаны, то возможны эффективные средства решения соответствующих задач. Работы по моделированию «миров» в настоящее время выполняются на базе создания программных комплексов. В то же время было бы целесообразно развернуть работы по созданию специальных БИСов и микропроцессоров для роботов и нейроподобных сетей. При этом уже сейчас следует обращать внимание на задачи унификации, классификации и прогнозирования компонентов БИСов и микропроцессоров.

Что касается работ по математическим методам, то я не буду их классифицировать, но скажу лишь, что нам очень нужны все методы, которые у нас развиваются: оптимизационные, методы анализа надежности, моделирования больших систем, численные методы решения уравнений и др.

В работах по методам должны достигаться две цели: высокий мировой уровень работ, который находит соответствующую объективную оценку, и создание соответствующих пакетов программ, т. е. реализация разработанных математических средств на ЭВМ. Пакетное оформление методов чрезвычайно важно.

Я перечислил далеко не все задачи, которыми наши молодые исследователи могли бы плодотворно заниматься. Будем надеяться, что они с присущим им молодым задором, настойчивостью и нерастраченной энергией продолжают поиски новых путей развития кибернетической науки и ее приложений в народном хозяйстве на благо нашей Отчизны.

ПРЕЕМСТВЕННОСТЬ ПОКОЛЕНИЙ ЭВМ

(Экономическая газета.— № 24, июнь 1972 г.)

На современном этапе экономического развития управление становится одним из важнейших средств повышения темпов научно-технического прогресса. В девятой пятилетке необходимо решить большие задачи в области создания автоматизированных систем управления. К ним относятся создание и широкое распространение автоматизированных систем управления предприятиями, технологическими процессами, создание отраслевых автоматизированных систем управления, и наконец, поставлена грандиозная задача создания общегосударственной автоматизированной системы сбора и обработки информации.

В области электронной вычислительной техники поставлена цель разработать в предстоящей пятилетке научные основы и создать такой научный и конструкторский задел для конструирования вычислительных машин четвертого поколения, чтобы в следующей пятилетке перейти к их серийному выпуску.

Ближайшая практическая задача — обеспечение выпуска в достаточном количестве вычислительных машин, периферийного оборудования, устройств сопряжения периферийного оборудования с ЭВМ, устройств, обеспечивающих связь ЭВМ, удаленных на значительное расстояние, а также соответствующего математического обеспечения.

Основой создаваемых в настоящее время АСУ является ЭВМ «Минск-32». В ближайшее время ожидается появление ЭВМ серии ЕС ЭВМ. Промышленность начинает массовый выпуск машин третьего поколения.

Нас беспокоит, что некоторые машины третьего поколения при той же пропускной способности периферийных устройств (периферии), что и «Минск-32», будут стоить в несколько раз дороже. Нам нужны машины третьего поколения для решения задач, но до тех пор, пока комплексные мероприятия не обеспечат сопоставимые цены, должны широко использоваться машины второго поколения.

Для ЕС ЭВМ разрабатывается большое количество вспомогательных устройств. В частности, по периферийной технике номенклатура таких устройств насчитывает более 20 наименований. Целесообразно выбрать небольшое количество типов регистраторов и ускорить организацию их серийного выпуска.

Относительно запоминающих устройств, используемых в ЭВМ, отметим, что отсутствие памяти на дисках, в частности, для ЭВМ «Минск-32», существенно снижает возможности создания эффективных систем управления на базе этой машины. Хранение системного математического обеспечения на магнитных лентах сильно снижает быстродействие систем. Дисковая память необходима также для организации хранения оперативных массивов банка данных. В связи с этим решение вопроса подсоединения к ЭВМ «Минск-32» дисков крайне необходимо.

Учитывая, что наряду с ЭВМ серии «Минск» появятся ЭВМ серии ЕС ЭВМ, чрезвычайно важно обеспечить их совместимость на уровне устройств сбора информации и внешней памяти. Однако этот вопрос решается пока медленно.

Одной из основных составляющих АСУ является ее информационный банк — хранилище информации. Большое разнообразие, а также большой объем информации в АСУ требуют наличия в вычислительной машине значительного и сложного программного аппарата ведения информационного банка: приема, хранения, размещения и выдачи информации по требованию.

Как показывает опыт, первоначальный этап создания банка является чрезвычайно трудоемким и ответственным делом. Естественно, что перестройка информационного банка крайне нежелательна и может послужить тормозом совершенствования АСУ.

Сотни организаций приступили к созданию АСУ, а значит, и информационных банков на базе ЭВМ «Минск-32». В ближайшие год-два работы по созданию соответствующих банков будут близки к завершению на большинстве таких объектов. Примерно к этому времени появятся машины серии ЕС ЭВМ. Ясно, что безболезненный переход на новую техническую базу невозможен без соблюдения принципа информационной преемственности.

Как решать эту задачу?

Во-первых, нужна стандартизация информации, используемой в АСУ. Во-вторых, нужна стандартизация машинного представления информации. В-третьих, в случае невозможности организации обмена информацией в АСУ в соответствии со стандартами машинного представления необходимо снабдить соответствующий технический комплекс и его математическое обеспечение аппаратом, позволяющим исключить имеющиеся расхождения в представлении информации.

Учитывая, что информационные банки для ЭВМ «Минск-32» создаются на магнитных лентах, целесообразно обеспечить стыковку этих машин с машинами ЕС ЭВМ на уровне магнитных лент и привязку математического обеспечения к выбранной структуре массивов.

Еще одной важной составляющей частью АСУ является ее программный фонд. Очевидно, что АСУ, которая в своей работе оперирует только первичной информацией, перестает быть управляющей системой и не представляет практического интереса. Эффективность АСУ, ее мощь и полезность зависят от набора программ по обработке первичной информации. Отметим, что накопление программного фонда осуществляется по мере формализации управления, связанной с построением математических моделей, отражающих различные стороны функционирования объектов управления, и решения ряда задач на базе этих моделей. Не затрагивая этот сложный вопрос, коснемся только процесса непосредственного программирования задач обработки данных.

Над созданием фондов работают большие коллективы квалифицированных программистов в течение длительного периода времени. Известно, что программирование на языке ЭВМ представляет собой трудоемкий процесс, в связи с чем появились алгоритмические языки. Программирование задач на таких языках позволяет не только ускорить сам процесс, но и получать программы, полностью не зависящие от конкретных ЭВМ, а значит, такие, которые могут быть использованы в других АСУ независимо от их технической базы. Учитывая это, можно значительно уменьшить дублирование работ при создании АСУ, освободив

этим большую часть квалифицированных специалистов для выполнения других работ.

При создании и развитии АСУ необходимо помнить, что один из важнейших принципов, касающихся математического обеспечения,— это программная преемственность АСУ на уровне алгоритмических языков. Обычно программы, включенные в АСУ, являются программами регулярного использования. Это предъявляет специальные требования к технике по быстродействию и объему занимаемой памяти. В то же время существующие и создаваемые трансляторы с алгоритмических языков либо полностью лишены аппарата оптимизации программ, либо он содержится в них «в зародыше». Это приводит к резкому снижению эффективности использования транслированных программ в АСУ. На первом этапе создания программных фондов выход из этого положения состоял в том, чтобы программировать задачи на языке ЭВМ или на языке, близком к языку ЭВМ, возлагая задачу оптимизации программ на программиста.

При создании систем программирования АСУ следует выделить задачу определения типовых или системных модулей обработки, из которых компонуются более сложные программы обработки, входящие в фонд. Решение этой задачи во многом зависит от конкретного объекта, для которого разрабатывается АСУ. Использование системных модулей при создании программного фонда, как показывает опыт, повышает производительность труда программиста по крайней мере в десять раз.

Составной частью математического обеспечения АСУ является ее операционная система — совокупность программных средств, назначение которых заключается в осуществлении управления ходом вычислительного процесса АСУ, причем существенным есть требование эффективности управления. Коротко задача состоит в следующем. Функционирование АСУ должно происходить в темпе (ритме) функционирования самого объекта управления. Прием данных и их обработка осуществляются при определенных условиях в зависимости от состояния объекта. Главная задача операционной системы —обеспечить соответствие между функционированием объекта и АСУ. При плохом управлении ходом вычислительного процесса операционная система не будет успевать за объектом и этим нарушит ритм управления. Операционная система, учитывая специфику объекта управления, должна содержать средства, исключающие возможность нарушения ритма.

Эффективное решение этой задачи можно получить путем создания аппарата прогнозирования действий на основе накопления истории функционирования объекта управления с тем, чтобы к моменту выполнения необходимых действий операционная система смогла выполнить все подготовительные операции. Создание такого аппарата является сложным, но возможным в связи с тем, что обычно процесс функционирования объекта является не случайным, а обладает определенной детерминированностью. Задача операционной системы заключается в выявлении этой детерминированности и использовании ее для повышения эффективности управления вычислительным процессом в АСУ. Первые шаги в этом направлении сделаны при развитии АСУП «Львов». Однако до полного решения данной задачи еще далеко.

Операционная система ЭВМ «Минск-32» не может быть использована при создании АСУ без существенных переделок, которые в настоящее время и осуществляются. Во избежание повторения ошибок предыдущих систем при создании операционной системы ЕС ЭВМ необходимо учесть все требования к ней со стороны разработчиков АСУ.

Важной задачей операционной системы является управление всеми устройствами технического комплекса АСУ. Сложность ее решения в том, что, во-первых, комплексность технической базы АСУ различна даже в АСУ одного типа, а, во-вторых, состав технических средств изменяется в связи с подключением новых устройств. Эти обстоятельства сказываются на операционной системе: ее приходится приспосабливать к технической базе. Необходимо создать генератор операционных систем. Получив исходную информацию — перечень устройств, входящих в комплекс технической базы, генератор формирует необходимую операционную систему. Такой подход еще раз подчеркивает необходимость соблюдения принципа модульности при создании систем математического обеспечения. Впервые отечественный опыт в решении данного вопроса получен при создании операционной системы для ЭВМ «Днепр-2».

Наконец, в качестве перспективной задачи в области операционных систем необходимо отметить задачу создания такой универсальной или стандартной системы сопряжения (интерфейса) математического обеспечения с вычислительными машинами, с помощью которой можно было бы генерировать программы не только применительно к архитектуре вычислительного комплекса, но и в зависимости от системы команд этого комплекса.

Необходимо организовать типовые разработки более высокого уровня, чем те, которые сейчас создаются. Речь идет о таких типовых решениях, которые отразились бы на работе по созданию технических средств и математического обеспечения. Прежде всего вопрос стоит о взаимоотношениях трех основных направлений: автоматизации технологических процессов, автоматизации проектно-конструкторских работ и автоматизации организационного управления. Единственно правильной политикой в этой области будет курс на гармоническое сочетание этих трех элементов.

В настоящее время большое значение имеют технологические системы управления, то же можно сказать о системе автоматизации проектирования. Системы организационного управления, помимо огромного прямого эффекта, создают предпосылки для правильного внедрения автоматизированных систем управления и других уровней проектирования и технологии. На данном этапе управление технологическими процессами страдает из-за отсутствия комплекса решений. Есть прокатный стан — нет соответствующих машин, и наоборот. А комплексное управление научными разработками — это одна из задач систем организационного управления. И если эти системы ориентировать на решение таких задач, то и работа по созданию АСУ технологическими процессами значительно ускорится.

ПЕРСПЕКТИВЫ АВТОМАТИЗАЦИИ ПРОЕКТИРОВАНИЯ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

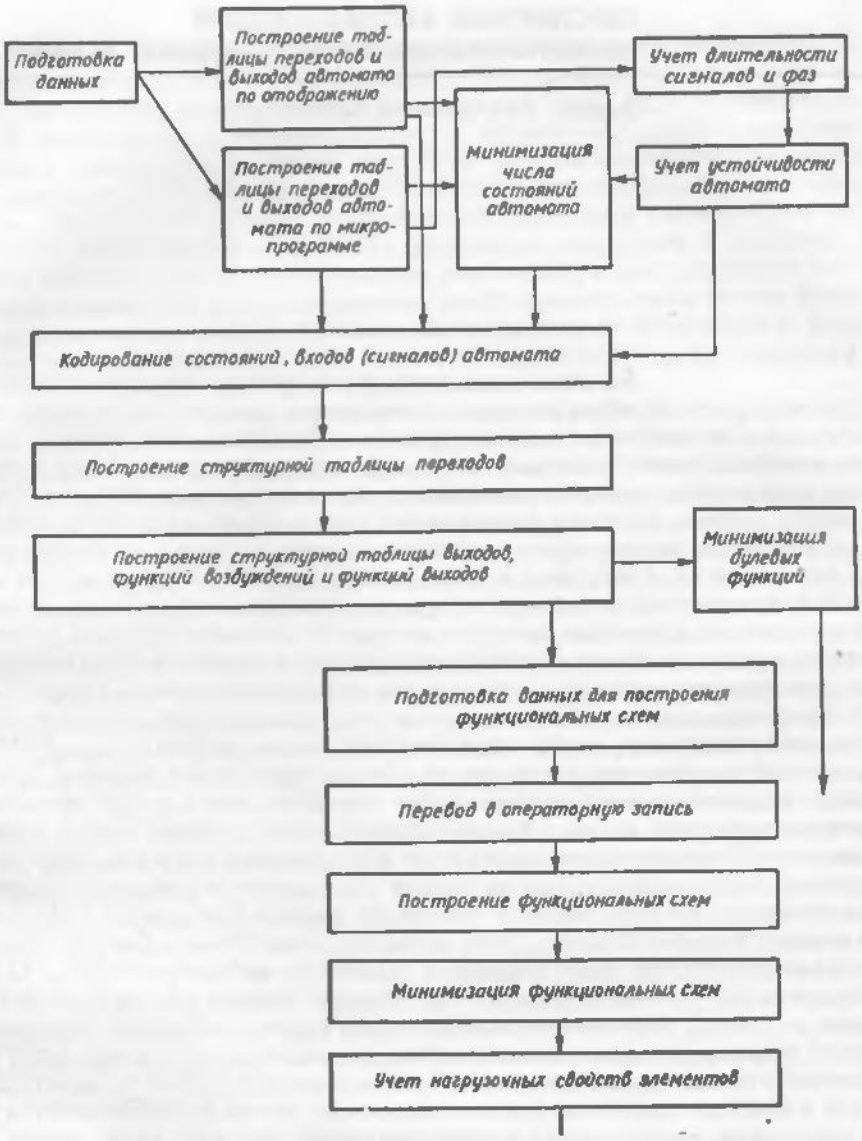
(Вестник АН СССР.— 1967.— № 4)

За последние годы резко ускорились темпы технического прогресса и одновременно усложнились различные технические конструкции, в том числе и конструкции вычислительных машин. Это, в свою очередь, резко увеличило нагрузку конструкторских бюро и проектных институтов, которые занимаются проектированием тех или иных объектов и систем. Поэтому в настоящее время очень актуальна задача автоматизации процессов проектирования во всех областях — не только в вычислительной технике, но и в машиностроении и приборостроении вообще. В ряде отраслей технического проектирования создаются программы или даже системы программ, которые позволяют автоматизировать отдельные этапы проектирования, а иногда осуществлять комплексную автоматизацию его процессов, что резко поднимает производительность труда.

Вопрос проектирования вычислительных машин стоит особенно остро. Вовьмем, например, такую сравнительно малую машину, как «МИР». Техническая документация для нее по объему превышает размеры самой машины. Подготовка этой документации отнимает очень много времени, что отодвигает сроки ввода в производство и эксплуатацию вычислительной машины. Поэтому в Советском Союзе и за рубежом относительно давно начались работы по различным аспектам автоматизации проектирования вычислительных машин. В нашей стране эти работы проводятся в Москве, Ленинграде, Томске, Минске, Риге, Ереване, Севастополе, Свердловске и других городах, в том числе в Киеве, в Институте кибернетики АН УССР.

Существуют два подхода к автоматизации проектирования вычислительных машин. В ряде организаций велись и ведутся работы по созданию отдельных программ для автоматического решения тех или иных задач на различных этапах проектирования. Однако начиная с 1962 г. наметился переход к комплексным системам автоматизации, когда разрабатывается не одна программа, позволяющая автоматизировать тот или иной расчет, а целый набор взаимосвязанных программ, осуществляющих комплексную автоматизацию целых этапов проектирования.

На рисунке показана блок-схема диспетчера так называемой малой системы проектирования, созданной в Институте кибернетики. Эта система в настоящее время используется не только в институте, но и в других организациях. Каждый блок ее соответствует программе. Общее количество приказов системы — 25 тыс. Среднее время работы каждой программы — 5—7 мин. Система дает возможность полностью автоматизировать процесс проектирования цифровых автоматов, для которых произведение числа входных сигналов на число состояний не превышает 10^3. Такие автоматы соответствуют относительно небольшим блокам вычислительной машины (порядка 12 триггеров).



Для малой системы автомат задается на одном из специальных языков (например, в виде микропрограммы), а результатом ее работы являются функциональные схемы заданного автомата, т. е. описание связей между элементарными автоматами, из которых конструируется автомат. В качестве элементной базы используется стандартная потенциальная система элементов. Переход от исходного задания автомата к функциональным схемам осуществляется в несколько этапов, причем на каждом этапе решается некоторая задача оптимизации.

В настоящее время задача автоматизации проектирования сильно усложнилась в связи с новыми особенностями современных вычислительных машин. Например, появилась так называемая система разделения времени, превращающая вычислительную машину в сложную систему об-

работки данных. Развиваются алгоритмические языки, которые, в свою очередь, требуют расширения возможностей внутренней логики машины и усложнения ее конструкции (различного рода системы трансляции, интерпретации и т. д.). Становятся более сложными системы команд.

Учитывая эти обстоятельства, проектирование современных вычислительных машин можно представить в виде процесса, состоящего из нескольких основных этапов (см. таблицу).

На этапе системного проектирования определяется количество блоков памяти и других устройств вычислительной машины, изучаются потоки заявок в вычислительном процессе проектируемой машины, выбирается такая организация вычислительного процесса, которая обеспечивает максимальную производительность устройств и всей машины в целом. В ходе

Этап	Задача	Результат	Язык
Системное проектирование	Разработка общей структуры машины, методов разделения времени, стратегии диспетчера и т. д.	Общая блок-схема, параметры блоков и характер их взаимодействия	СЛЭНГ
Алгоритмическое проектирование	Разработка вопросов трансляции, интерпретации, системы команд и т. д.	Алгоритмы функционирования блоков, описанных на языке АЛОС	АЛОС
Блочное проектирование	Построение блок-схемы, выбор состава устройств микроопераций, оптимизация алгоритмов, разделение на управляющие и операционные блоки, внутренняя синхронизация и т. д.	Физическая блок-схема	АЛОС
Автоматное проектирование	Структурный синтез автоматов. Получение технической документации	Система булевых функций	Язык микропрограммы Язык булевых функций
Техническое проектирование	Построение функциональных и монтажных схем. Получение технической документации	Техническая документация для изготовителей машины	Язык функциональных схем

алгоритмического проектирования составляются основные алгоритмы, характеризующие функционирование отдельных устройств проектируемой машины. Блочный этап — это разработка подробной блок-схемы вычислительной машины, допускающей дальнейшее проектирование каждого блока в отдельности. Исходным заданием для этого этапа является алгоритмическое описание вычислительной машины. На автоматном этапе каждое отдельное устройство (блок) проектируется с применением методов малой системы автоматизации. В процессе технического проектирования решаются задачи разбиения функциональной схемы на модули и платы, задачи расчета отдельных элементов функциональной схемы и построения монтажных схем.

Сейчас в Институте кибернетики АН УССР разрабатывается большая автоматизированная система проектирования, охватывающая все перечисленные этапы и включающая малую систему в качестве автоматного этапа.

Отметим, что системный и алгоритмический этапы вследствие возникающих здесь трудностей пока автоматизированы лишь частично.

Каковы же задачи, появляющиеся при автоматизации проектирования? Основная задача — создание формальных языков для описания структуры вычислительной машины на каждом этапе проектирования. Но не одних только формальных языков. Необходимы также трансляторы, которые переводят запись на языке высшего уровня в запись на языке низшего уровня, например с языка системного на язык алгоритмов, с языка блочного описания на язык автоматов и, наконец, на язык монтажных схем. При этом каждый раз добавляется информация о способах реализации, скажем, о системе элементов проектируемой машины. В то же время для каждого этапа составляется соответствующая документация. Если на верхнем уровне описание проектируемой машины сравнительно короткое и занимает несколько страниц текста на входном языке, то затем оно разрастается в огромные тома технической документации. Все описания на соответствующих формальных языках должны проводиться автоматически с помощью систем транслирования.

Однако формальных языков и систем транслирования недостаточно для автоматизации проектирования. Это связано с тем, что на каждом уровне в рамках данного языка можно осуществлять формальные преобразования с сохранением тех или иных инвариантов, но с изменением других параметров конструкции, так, чтобы получить наилучшие технические решения на определенном этапе проектирования. Формальные преобразования в языках требуют развития различных математических областей — от теории больших систем и соответствующих преобразований в рамках этой теории на верхнем уровне до преобразования монтажных схем на нижних уровнях.

Теперь охарактеризуем проблемы, связанные с нахождением критериев и путей оптимизации, так как формальные преобразования надо делать не вообще, а с тем, чтобы достичь тех или иных конкретных целей. Если на каком-то этапе задача проектирования настолько сложна, что полная оптимизация пока еще невозможна, производится частичная оптимизация либо проектируемая машина моделируется и организуется совместная работа человека с моделирующей машиной, позволяющая быстро оценить ряд вариантов, осуществить формальное преобразование схем в языке и получить лучшие характеристики схем на данном этапе проектирования применительно к тем или иным локальным критериям оптимальности.

Каков же характер основных языков на разных этапах проектирования и как обстоит дело с решением различного рода научных задач в отношении формальных преобразований, т. е. грамматики и алгебры языков?

На двух верхних этапах — в системном и алгоритмическом проектировании — используются различного рода специализированные в том или ином направлении, но универсальные с точки зрения алгоритмических возможностей языки.

Известно, что задачей вычислительной машины как устройства для переработки дискретной информации является преобразование входных последовательностей дискретных сигналов, скажем, букв, в выходные последовательности. Машина осуществляет отображение множества входных последовательностей сигналов в множество выходных последовательностей сигналов. Это аналог функции. Обычно имеются определенные конструктивные ограничения, накладываемые на реализацию функции.

Это связано с тем, что далеко не всякое отображение можно осуществить в машине непосредственно за один шаг. Необходимо разложить это отображение так, чтобы получить композицию более мелких отображений. Выделяя различным образом более мелкие преобразования в качестве отдельных элементов, можно строить разные алгоритмические системы. Процесс разложения на элементарные отображения называется алгоритмированием.

В случае описания машины с помощью различных алгоритмических языков, используемых на алгоритмическом и блочном этапах проектирования, инвариантом является отображение. Это означает, что отображение не должно изменяться. При преобразованиях в данных языках изменениям подвергается реализация отображения, т. е. способ его разложения на элементарные составляющие части. Таких способов существует достаточно много, и необходимо уметь осуществлять формальные преобразования от одного способа к другому.

На следующем этапе используется язык теории автоматов. На автоматном уровне инвариантом служит не только само отображение (задаваемое машиной), но и способ его разложения на составляющие. Работа автомата задается с помощью двух функций:

где $a(t)$ — состояние автомата в момент t; $x(t)$ — входной сигнал в момент t; $y(t)$ — выходной сигнал в момент t.

Функция перехода

$$a(t) = f(a(t-1)),\ x(t),\ t = 1,\ 2,\ 3,\ \ldots$$

Функция выходов

$$y(t) = \varphi(a(t-1),\ x(t)),\ t = 1,\ 2,\ 3,\ \ldots$$

Автомат имеет множество внутренних состояний. Эти состояния изменяются в дискретные моменты времени. Существует система выходных сигналов. Следующее состояние автомата — функция от предыдущего состояния и от входного сигнала, выходной сигнал — функция от тех же аргументов. Эти функции, называемые функциями переходов и выходов, являются объектами для преобразований на автоматном этапе проектирования.

Наконец, на последующих уровнях проектирования рассматриваются автоматы частного вида, имеющие единственное внутреннее состояние. В этом случае инвариантом служит выходная функция соответствующего автомата, она не должна изменяться при формальных преобразованиях. Формальные преобразования меняют способ реализации этих функций. Наконец, на уровне монтажных схем язык строится очень просто. Если это обычные монтажные схемы, неинтегральные, то элементами языка являются системы чисел, координат, описывающих положение той или иной точки на плоскости (положение узлов соответствующих схем). Инвариантами служат, во-первых, топология (граф) соединения узлов и, во-вторых, расстояния между некоторыми узлами, а преобразования — это изменения координат узлов. Одним из критериев эффективности может быть достижение минимальной длины всех соединительных проводов. Этот критерий зависит от рода выбранных элементов.

На уровне монтажных схем научные проблемы относительно языка этих схем в классическом виде по существу не стояли. Здесь речь идет в

основном о том, чтобы рациональным образом стандартизировать обозначения. В микроэлектронике возникают дополнительные проблемы. Но тем не менее вопросы, связанные с построением формальных языков и формальными преобразованиями в таких языках, достаточно просты. А вот вопросы оптимизации очень сложны на всех этапах, в частности и на этом этапе, так как они сводятся к математическим задачам нахождения максимума или минимума функций от очень большого числа аргументов и часто не ограничиваются задачами линейного программирования. Соответствующие параметры исчисляются многими десятками или сотнями. Поэтому решение такого рода задач затруднительно и требует достаточно больших программ.

Что касается языка функциональных схем, то это по сути язык алгебры логики. Он был создан задолго до появления вычислительных машин — в середине прошлого столетия. Формальные преобразования в рамках этого языка тоже развивались на протяжении многих лет, но только начиная с 1938 г. они стали применяться для синтеза различного рода дискретных схем.

Задачи оптимизации на этапе функциональных схем довольно сложны, так как язык математической логики не учитывает специфики этих задач. Например, при построении функциональной схемы надо соблюдать некоторые ограничения, связанные с количеством разветвлений схемы, что в классической математической логике, естественно, не учитывается. Еще одна трудность состоит в том, что первичными элементами не обязательно являются те, которые соответствуют элементарным операциям классической математической логики. Выбор элементов зависит от уровня развития радиоэлектроники, микроэлектроники, особенностей различного рода схем, которые можно сравнительно просто изготовить. Эти обстоятельства усложняют как теорию преобразования, так и задачи оптимизации.

Рассмотрим более подробно системный и алгоритмический этапы проектирования.

При проектировании той или иной вычислительной машины прежде всего необходимо знать задачи, которые предполагается решать с помощью проектируемой машины.

Современная система обработки данных имеет большое число периферийных пультов, которые связаны с центральной машиной и с малыми вычислительными машинами. Если задачи таковы, что не могут быть решены с помощью малой машины, данные передаются на соответствующем языке в центральную машину. Для этого существует специальное устройство, которое обеспечивает подключение различных видов оборудования (записывающих устройств, большого числа оперативных памятей, магнитных барабанов и т. д.).

Главная задача на данном этапе состоит в том, что надо разработать программу диспетчеризации. При этом возникают вопросы: каким образом программа-диспетчер должна удовлетворять запросам потребителей, задающих, например, с пятидесяти различных мест пятьдесят разных задач, как организовать внутреннюю связь с машиной, с системой расположения информации в запоминающих устройствах, чтобы обращение к ней происходило в возможно более короткое время, чтобы возможно меньше простаивало оборудование?

Для исследования этой системы на первом этапе составляют модели будущей машины, а также каждого отдельного устройства. Модели изучаются на действующей вычислительной машине. Испытывается большое

число вариантов. Исследуется, как организовать управление вычислительной машиной, какими будут характеристики системы, как будут удовлетворяться запросы и т. д.

В настоящее время построение индивидуальных моделирующих программ каждой новой машины — пройденный этап. Существует возможность создать универсальную систему моделирования, в которой будут использоваться языки для системного описания. Таких языков разработано много: SOL, SIMULA, SIMSCRIPT и т. д. Смысл этих языков состоит прежде всего в том, что они направлены не на задачи, не на потребителя, а на систему. В них есть специальные средства для описания потоков заявок, различных процессоров, с которыми приходится иметь дело при проектировании вычислительных машин, а также средства для моделирования системного времени и построения гистограмм распределений величин, получаемых в результате моделирования. При этом широко используются другие особенности, например средства списочного языка и пр.

Не могу охарактеризовать все языки. Мы пользуемся в Институте кибернетики языком СЛЭНГ, который является вариантом языка SOL. СЛЭНГ дополнен рядом новых элементов по сравнению с SOL.

Как решается задача? На языке СЛЭНГ соответствующий комплекс описывается сравнительно просто:

ПРОГРАММА МОДЕЛИРОВАНИЯ ПРОЦЕССА
ФУНКЦИОНИРОВАНИЯ ПУЛЬТОВ — ПРОЦЕССОРОВ —
БАРАБАНОВ — ОЗУ

```
начало целый a, b, c, d, e, f, h, s, l, k, F [a], G [b], H [c], Ф [d], P [e],
Q [f], R [h]
действительный r
устройство ПРОЦЕССОР [2], БАРАБАН [k]
память ОЗУ [2], S, ПУЛЬТ, l
таблица ТАБ (100 шаг до 50 000)
ввод (a, b, c, d, e, f, h, s, l, k, r, F, G, H, Ф, P, Q, R)
```

```
процесс ПОЛЬЗОВАТЕЛЬ (КВ, ПРОЦ, ОБЪЕМ)
      Целый КВ, ПРОЦ, ОБЪЕМ
начало
М: новый ПОЛЬЗОВАТЕЛЬ (распред (F), распред (Q), распред (G)) на М1
   ждать распр д (H); на М
М1: войти ПУЛЬТ; набор
М2: новый КВАНТ (распред (Ф), ПРОЦ, ОБЪЕМ, время) на ОБСЛ
М5: объединить 2; КВ : = КВ — 1,
      если КВ ≠ 0 то начало ждать распред (P) p
            на М2 конец
      выйти ПУЛЬТ
      конец ПОЛЬЗОВАТЕЛЬ
```

```
процесс КВАНТ (ВО, НПР, П, Т) целый ВО, НПР, П, Т
начало
ОБСЛ: удалить ОЗУ [НПР], П
            новый ОБМЕН (ПО, выбор (1, k)) на Б
```

```
M3: объединить 2
    занять ПРОЦЕССОР [НПР]
    a := если q > ВО то ВО иначе q; ВО := ВО — a
    ждать a, освободить ПРОЦЕССОР [НПР]
    новый ОБМЕН (П, 1, выбор (1, k) на Б
M4: объединить 2
    выйти ОЗУ [НПР], П
    если ВО ≠ 0 то на ОБСЛ
    табулировать время — Т в ТАБ на М5
    Конец КВАНТ
```

```
процесс ОБМЕН (МАССИВ, НАПР, НБ)
        целый МАССИВ, НАПР, НБ
начало удалить
Б: занять БАРАБАН [НБ]
   ждать [распред (ВП) + МАССИВ × r
   освободить НБ
   если НАПР = 0 то на М3, иначе на М4
   конец ОБМЕН
```

КОНЕЦ

КОММЕНТАРИИ

S	— объем ОЗУ в процессорах
l	— количество пультов
k	— количество барабанов
V	— время обмена одним словом с МБ
q	— квант времени, предоставляемый процессорами
F	— распределение числа взаимодействия с машиной в течение одного подхода к пульту
G	— распределение объема оперативной памяти, потребного пользователю
H	— распределение интервала между моментами поступления пользователей к моменту
Ф	— распределение требуемого времени процессора при каждом взаимодействии пользователя с процессором
P	— распределение промежутка времени между окончанием одного взаимодействия и началом следующего взаимодействия пользователя с процессором (распределение времени «думания» пользователя)
Q	— распределение номера процессора при входе пользователя в систему
R	— распределение времени поиска места на МБ

Процесс ПОЛЬЗОВАТЕЛЬ

КВ	— количество взаимодействий пользователя с процессором
ПРОЦ	— номер процессора
ОБЪЕМ	— объем ОЗУ, необходимый для помещения задачи пользователя

Процесс КВАНТ

ВО	— время обслуживания при одном взаимодействии
НПР	— номер процессора
П	— объем ОЗУ, необходимый для помещения задачи пользователя
Т	— момент начала взаимодействия

Процесс ОБМЕН

МАССИВ — количество слов в задаче пользователя
НАПР — направление обмена
НБ — номер барабана

В приведенном примере описаны три специальные программы: «Пользователь», «Квант» и «Обмен». С помощью законов распределения, которые задаются более или менее произвольно, моделируются распределения такого рода случайных событий, как подход пользователей к машинам, количество повторных запросов, объем задач, выражаемый суммарным объемом памяти, время работы процессора, требуемое для решения этих задач, место расположения информации на магнитном барабане или магнитной ленте.

Все это моделируется первой частью программы.

Затем вступает в действие следующая процедура — запросы передаются на центральный процессор, программа-диспетчер подключает оперативные запоминающие устройства и выделяет соответствующий объем памяти, если он свободен.

Последняя часть программы — обмен. Она характеризует обмен с внешней памятью на магнитном барабане и оперативной памятью процессора при решении задачи. Одновременно в этой программе предусмотрено табулирование времени ожидания пользователями начала взаимодействия с машиной.

После того как программа написана на языке моделирования, универсальный транслятор позволяет переписать ее в программу какой-либо действующей машины, например «БЭСМ-4», и моделировать эту систему, получая эмпирические распределения интересующих нас параметров. Вследствие того что рассматриваются параллельные процессы, приходится соблюдать много дополнительных ограничений. Непосредственно формальные преобразования пока не производятся. Частично задача формальных преобразований решается в рамках теории больших систем. Здесь же пока делается следующее: поскольку все, что записано в этих программах, достаточно наглядно, то, изменяя характеристики данных программ, удается быстро через систему трансляторов осуществить реальное моделирование в реальной машине.

Возникает вопрос, почему не пользоваться в этом случае известными закономерностями теории массового обслуживания, поскольку, в конце концов, мы имеем дело не с чем иным, как со специальной системой массового обслуживания.

Действительно, анализ показывает, что здесь почти ни одна достаточно общая задача не поддается аналитическим методам решения и не укладывается в схемы классической теории массового обслуживания. Главная трудность заключается в том, что существует обратная связь устройства для обработки данных с входным потоком заявок. В обычных классических задачах массового обслуживания поток заявок является независимым от обслуживающей системы. Здесь же в зависимости от состояния системы обработки данных меняются характеристики потока заявок. Кроме того, в процессе обслуживания формируются или просто включаются гораздо более сложные преобразования заявок входного потока. Потоки же являются неоднородными, и заявки несут в себе большое число параметров, от которых может зависеть процесс последующей обработки.

Тем не менее система автоматизации проектирования может охватывать множество отдельных задач, допускающих аналитическое решение.

Для этого в языке предусматриваются средства для описания стандартных потоков в системах массового обслуживания, а также соответствующих аналитических преобразований. Дело в том, что использование чистого метода Монте — Карло с применением такого языка требует в общем очень много времени и если есть возможность упростить модель за счет аналитического расчета отдельных ее частей, то это следует сделать.

Относительно же алгоритмического этапа отметим, что до последнего времени здесь по сути не было подхода к решению комплексных задач автоматизации проектирования. Языки создавались, но формальных преобразований в них почти не существовало. Вероятно, одной из первых работ, посвященных разработке данного вопроса, была работа Ю. И. Янова, который построил теорию формальных преобразований схем программ. В настоящее время эта работа достаточно популярна среди тех, кто занимается вопросами оптимизации. Однако в ней решалась только задача преобразования схемы алгоритма, а не задача преобразования самого алгоритма. Трудность последней состоит в том, что на заре теории алгоритмов была доказана алгоритмическая неразрешимость задачи установления эквивалентности двух алгоритмов с помощью формальных преобразований. Это «отпугивало» исследователей от построения такого рода формальных систем для преобразования алгоритмов. В последнее время удалось решить задачу построения математического аппарата преобразований специального вида алгоритмов микропрограмм, которую можно применять во многих практически важных случаях.

Кратко остановимся на основных научных идеях, используемых в этой теории. По существу всякий алгоритм есть не что иное, как способ задания преобразования некоторого множества. Для построения преобразований выбираются условия, заданные на этом множестве. На множестве преобразований (операторов) система операций строится, как в алгебре: перемножения операторов, α-дизъюнкции и α-итерации.

Пусть M — множество состояний автомата B, $a \in M$; $\mathfrak{A}$ — множество операторов, действующих на множестве M; $\mathfrak{B}$ — множество условий, определенных на множестве M. Операции α-дизъюнкции и α-итерации определяются следующим образом: $P, Q, R \in M$; α, $\beta \in \mathfrak{B}$ — дизъюнкция операторов P и Q есть оператор R, который преобразует элемент $a \in M$ в Pa, если условие $\alpha(a)$ выполняется и оператор P применим к a, и в Qa, если условие $\alpha(a)$ ложно и Q применим к a. В остальных случаях оператор R не определен.

Операция α-итерации определяется следующим образом: α-итерация оператора P есть оператор R, который преобразует элемент $a \in M$ в элемент $P^n a$, если не выполняются условия $\alpha(a)$, $\alpha(Pa)$, ..., $\alpha(P^{n-1}a)$, а $\alpha(P^n a)$ выполняется, в противном случае R не определено.

Такого рода операции использовались и ранее в различных логических языках. Вводится новая по сравнению с классическими операция умножения оператора на логическое условие. Результат умножения оператора P на условие α есть новое условие β, такое, что $\beta(a)$ выполняется тогда и только тогда, когда выполняется $\alpha(Pa)$.

В множестве операторов выделяются некоторые операторы, называемые элементарными или микрооперациями. Кроме того, фиксируются некоторые логические условия. Эти операторы и условия порождают алгебру микропрограмм, с помощью которой можно задавать алгоритмы, реализуемые устройствами проектируемой машины. Далее приводятся примеры некоторых микроопераций и условий, используемых при записи алгоритма проектируемой машины.

Пусть $(x_1, x_2, \ldots, x_n)$ — набор двоичных регистров. Состояниями регистров являются рациональные числа, записанные в двоичной системе счисления. Рассмотрим микрооперации и условия:

$$l_i: \quad x_i := 2x_i \qquad (i = 1, 2, \ldots, n),$$

$$r_i: \quad x_i := \frac{1}{2} x_i \qquad (i = 1, 2, \ldots, n),$$

$$S_{ij}: x_j := x_i + x_j \qquad (i, j = 1, 2, \ldots, n),$$

$$P_i: \quad x_i := x_i + 1 \qquad (i = 1, 2, \ldots, n),$$

$$P_i^{-1}: x_i := x_i - 1 \qquad (i = 1, 2, \ldots, n),$$

$$O_i: \quad x_i := 0 \qquad (i = 1, 2, \ldots, n),$$

e — тождественное преобразование; $\alpha_i = $ И, если $x_i = 0$, иначе Л $(i = 1, 2, \ldots, n)$; $\beta_i = $ И, если $x_i = 2k$, иначе Л (k — целое, $i = 1, 2, \ldots, n$).

Отличие данной теории от классической теории алгоритмов состоит в том, что алгоритмы рассматриваются как элементы некоторой алгебры и для их преобразования можно применять соотношения соответствующей алгебры.

Соотношения в алгебре микропрограмм гораздо сложнее и их значительно больше. Приведем некоторые примеры таких соотношений:

$$l_i S_{ij} = S_{ij}^2 l_i \quad (i, j = 1, 2, \ldots, n),$$

$$Z_i O_i = O \quad (Z_i : x_i := f(x_i), \quad i = 1, 2, \ldots, n),$$

$$r_i P_i = P_i^2 r_i \quad (i = 1, 2, \ldots, n),$$

$$l_i \alpha_i = r_i \alpha_i = \alpha_i \quad (i = 1, 2, \ldots, n),$$

$$y_i Z_j = Z_j y_i : (y_i : x_i := f(x_i);\ Z_j : x_j := \varphi(x_j);\ i \neq j),$$

$$\{(yP_i^{-1}Z)^s\}\,\alpha_i = \beta_i, \text{ где } \mu = \alpha_i \vee P_i^{-1}\alpha_i, \text{ а } y\alpha_i = Z\alpha_i = \alpha_i \quad (i = 1, \ldots, n),$$

$$\{P\}_{\mu} = (e \vee_{\nu} P)\{P^2\}, \text{ где } P \in \mathfrak{A},\ \mu \in \mathfrak{B} \vee = \{P^2\}_{\mu \vee P^{\mu}},$$

$$\{P\}_{\mu} Z = Z\{N\}, \quad \text{где } P, Q, Z \in \mathfrak{A},\ P,\ Z = ZQ \text{ и } Z_{\mu} = \mu.$$

Рассмотрим пример преобразования алгоритма умножения двух целых чисел. Пусть x_1, x_2, x_3 — целые числа. Начальная запись этого алгоритма соответствует определению алгоритма умножения как последовательного сложения. Оказывается возможным преобразовать эту запись в такую, которая соответствует алгоритму, обычно реализуемому в вычислительной машине:

$$Q: x_2 := x_1,\ x_3, \quad x_1 := 0;\ x_3 := 0,$$

$$Q = O_2 \{S_{12}P_3^{-1}\}_{\alpha_3} O_1 O_3,$$

$$Q = O_2 \{S_{12}P_3^{-1}\}_{\alpha_3} l_1 r_3 O_1 O_3,$$

$$Q = O_2 (e \vee_{\beta_3} S_{12}P_3^{-1})\{S_{12}^2 P_3^{-2}\}_{\alpha_3} l_1 r_3 O_1 O_3,$$

$$Q = O_2 (e \vee_{\beta_3} S_{12}P_3^{-1})\, l_1 r_3 \{S_{12}P_3\}\, O_1 O_3,$$

$$Q = O_2 ((e \vee_{\beta_3} S_{12}P_3^{-1})\, l_1 r_3)^k \{S_{12}P_3^{-1}\}_{\alpha_3} O_1 O_3,$$

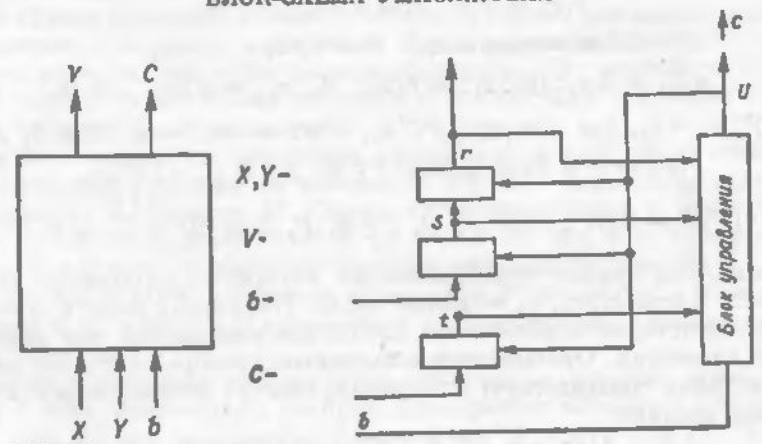
$$Q = O_2 \{(e \vee_{\alpha_3 \beta_3} S_{12}P_3^{-1})\, l_1 r_3\}\, O_1 O_3.$$

Практический опыт использования идеи преобразования алгоритмов как выражений в микропрограммных алгебрах свидетельствует о том, что это средство является достаточно эффективным и дает возможность формализовать преобразование структуры проектируемой машины в целом на алгоритмическом этапе. Оно может быть применено и к преобразованию программ для вычислительной машины. Правда, не следует думать, что здесь все задачи решены и все очень просто, так как вопросы, касающиеся преобразования программ, связаны с огромным количеством соотношений. До практической реализации придется пройти большой этап преодоления реальных технических трудностей программирования.

Кстати, отметим, что ощущается острая необходимость в вычислительных машинах, для которых естественными являются операции преобразований буквенных выражений, скажем, операция машины, а не специальные программы, компоновка которых занимает значительно больше времени, чем это требуется для решения задачи. Такая машина создается в Институте кибернетики АН УССР. Она, правда, не полностью решает проблему, так как для ее решения требуются машины гораздо более высокого быстродействия — самого высокого, которое достигнуто в настоящее время.

На базе теории преобразований алгоритмов построен специальный язык АЛОС для описания алгоритмов функционирования устройств вычислительной машины. Приведем описание на языке АЛОС устройства, которое выполняет умножение двух n-разрядных чисел.

БЛОК-СХЕМА УМНОЖИТЕЛЯ



ПЕРЕМЕННЫЕ
входные X, Y (n), b (1)
выходные V $(n+1)$, c (1)
внутренние r, S, t (n)
константа десятичная n
функция сдвиг (a, v)
сдвиг $[1]$ = если $v = 1$, то 0, иначе a $[2]$
сдвиг $[i]$ = если $v = 1$, то a $[i-1]$, иначе a $[i+1]$, $2 \leqslant i \leqslant n-1$
сдвиг $[n]$ = если $v = 1$, то a $[n-1]$, иначе 0
функция сдвиг — влево (a) = сдвиг $(a, 0)$
функция сдвиг — вправо (a) = сдвиг $(a, 1)$
функция сумма — (a, b) вспомогательная g
$g[n+1] = 0$
$g[i] = a[i+1] \wedge b[i+1] \vee a[i+1] \wedge g[i+1] \vee b[i+1] \wedge g[i+1]$, $1 \leqslant i \leqslant n$
сумма $[i] = a[i] \neq b[i] \neq g[i]$, $1 \leqslant i \leqslant n+1$

```
начало
  L1 : если b = 0, то на L1
  S : = x, t : = y, r : = 0 [n + 1]
    L2: если t [n] = 1, то r : = сумма (r, 0 • S)
  t : = сдвиг — вправо (t)
       если t = 0 (n), то на L3
       если S [1] = 1, то (r [1] : = 1, на L3) L3 : C : = 1,
       на L1
          КОНЕЦ
```

Как автомат практически оно не может быть описано, поскольку количество состояний этого автомата столь велико, что никакая реальная программа не может его эффективно преобразовать. АЛОС позволяет проектировать и такие автоматы, так как способ задания функций и самих элементов на этом языке дает возможность применять технику преобразования алгоритмов к преобразованию автоматов. Сейчас эти преобразования делаются формально пока вручную. Существует метод формального разделения проектируемой машины на блоки, которые также описываются на языке АЛОС.

В дальнейшем все это будет реализоваться в виде системы программ, что представляет довольно сложную задачу.

Таким образом, отличительная черта нынешнего этапа автоматизации и проектирования состоит в том, что начинает создаваться математическая база перехода к более или менее полной автоматизации процессов проектирования. Операционное устройство с устройством управления, иными словами, процессор вычислительных систем, может быть описано и спроектировано формально. Если должным образом организовать работу, то можно надеяться, что на протяжении 3—4 лет будут созданы программы, которые дадут возможность решить эти задачи. Но до полной автоматизации проектирования вычислительной машины еще далеко. Здесь еще многое требуется уточнить.

Дело в том, что очень сложной является проблема автоматизации проектирования системы математического обеспечения вычислительной машины.

Современная вычислительная машина — не просто собрание электронных приборов. Машиной сейчас правильно называть только машину с системой математического обеспечения. В состав ее нужно включать, по крайней мере, внутреннее математическое обеспечение, т. е. всевозможные средства трансляции, диспетчеры, различного рода системы алгоритмов, языки и т. д.

Введя небольшие конструктивные изменения в структуру машины, можно сильно упростить трансляторы. Но это трудно сделать, когда трансляторы создаются к готовой машине. Поэтому такого рода работы обязательно нужно проводить вместе с проектированием машины, и она должна выпускаться с математическим обеспечением.

Составление систем сравнительно простых программ можно выполнить с привлечением средств автоматического программирования и формальной оптимизации одновременно с проектированием машины. Когда же речь идет о большой машине, трудности возрастают.

Относительно внешнего математического обеспечения единой точки зрения нет. Американская фирма «IBM», чтобы сохранить систему внешнего обеспечения для своих машин, встроила в новую машину систему интерпретации программ, написанных на языке старых машин. Однако такая система не оправдала себя. Поэтому все-таки необходимо осуществлять

перетранслирование программ, написанных на старых машинах, в новые программы.

По-видимому, наиболее целесообразен путь, при котором создаются система математического обеспечения, написанная на разных языках типа ФОРТРАН, АЛГОЛ и других, и универсальный транслятор для перевода на языки новых машин.

Трудность здесь состоит в том, что современные системы транслирования чаще всего не обладают хорошими средствами оптимизации, поэтому качество программ, составленных с помощью транслирования, значительно хуже программ, сделанных вручную. Необходима дальнейшая работа в области построения формальных методов преобразования программ в рамках изложенной здесь теории преобразования алгоритмов. В принципе это возможно, но требуются слишком большие вычислительные мощности. Видимо, нужно разрабатывать эмпирические промежуточные методы, которые позволяют приблизить уровень качества программ, сделанных методом транслирования, к ручным программам.

Чрезвычайно привлекательным представляется создание модели или системы, с помощью которой можно автоматизировать не только проектирование, но и процесс изготовления вычислительных машин. До настоящего времени такого рода возможности являлись предметом исследования, но реальных шансов на то, что они осуществимы, не было. Частично автоматизация процесса была внедрена для навесных деталей в некоторых зарубежных фирмах. В связи с развитием микроэлектроники и интегральных схем открывается принципиальная возможность соединить задачу автоматического проектирования с задачами изготовления вычислительных машин. Правда, современная техника, которая пользуется интегральными схемами (выращиванием монокристаллов, диффузией, высокотемпературными процессами), настолько качественно разнообразна, что полная автоматизация здесь представляется довольно затруднительной. Однако в последние годы начинает развиваться новая технология схем микроэлектроники, основанная на принципе электроники, когда вся обработка схемы происходит в вакууме и представляет собой один тип процесса — обработку электронным и ионным лучами.

В настоящее время такого рода опытные установки созданы, и в ряде мест разрабатываются специальные машины, которые позволяют автоматизировать процесс изготовления схем, в том числе многослойных, методами электроники. Есть основание надеяться, что совместная работа в этом направлении специалистов, занимающихся различного рода вопросами автоматизации проектирования, и специалистов в области микроэлектроники приведет к созданию достаточно эффективных систем автоматизации. Эти системы позволяют резко сократить время проектирования машин, повысить их сложность в связи с переходом на новую технологию и, наконец, резко их удешевить.

МАТЕМАТИЧЕСКИЕ АСПЕКТЫ АВТОМАТИЗАЦИИ ПРОЕКТИРОВАНИЯ ЭВМ

(I Всесоюз. совещ. по автоматизации проектирования в машиностроении. Минск.— 1978)

Основными особенностями развития и внедрения автоматизации проектирования ЭВМ являются превалирование системного подхода и появление индустрии средств обработки информации.

Под системным подходом, реализуемым в автоматизированных системах проектирования ЭВМ, подразумевается интегрированное рассмотрение и представление в системе как объектов, так и операций проектирования на различных стадиях процесса проектирования. Более того, в этих решениях должны находить отражение не только существующие технологии проектирования, но и их развитие в будущем. Создание индустрии средств обработки информации требует качественно нового уровня унификации и типизации решений по техническому, математическому, информационному и организационному обеспечению систем, а это непосредственно связано с использованием математического аппарата в процессе проектирования.

Рассмотрим некоторые математические вопросы, возникшие при ручном и автоматизированном проектировании ЭВМ. Создание схемного и программного оборудования ЭВМ, разработка общего и специального математического обеспечения систем управления и систем проектирования требуют решения задачи проектирования алгоритмов. Теория проектирования алгоритмов занимается разработкой и исследованием их математических моделей, которые включают абстрактные модели общего характера и конкретные модели, связанные с определенными областями применения. В терминах тех или иных моделей исследуются следующие классы задач проектирования: анализ, синтез и оптимизация алгоритмов.

На базе результатов, полученных при исследовании основных задач проектирования алгоритмов, создаются практические методы их решения. Эти методы служат основой для создания технологий или методик проектирования алгоритмов и специальных систем математического обеспечения, которые используются в качестве средств автоматизации процессов проектирования.

Исследованные к настоящему времени модели алгоритмов можно разделить на алгебраические, динамические и структурные.

Первый класс моделей предназначен для описания функций, вычисляемых с помощью алгоритмов или преобразований информации. Обычно эти функции и преобразования выражаются алгебраическим образом через другие известные функции и преобразования, т. е. с помощью операций или уравнений, что определило название «алгебраические модели». Динамические модели используются для описания процессов преобразования информации, порождаемых алгоритмами. Наконец, структурные модели предназначены для описания структуры систем, реализующих задан-

ные наборы алгоритмов. Использование тех или иных видов моделей зависит от решаемых задач проектирования.

В общем случае процесс проектирования можно представить в виде серии построений некоторых цепочек моделей, в которых каждая следующая является реализацией предыдущих. Обычно в начале цепочки рассматриваются алгебраические модели, затем динамические, и, наконец, окончательный результат представляется в виде структурной модели. Одной из задач математической теории проектирования является определение алгоритмов перехода от одних моделей к других.

Алгебраические модели. Основой для их построения служит алгебра данных, которую можно рассматривать как универсальную Ω-алгебру. Конкретные алгебры данных обычно связываются с языками программирования или с некоторыми классами задач, как, например, алгебра данных языка АЛГОЛ или алгебра данных в задачах анализа свойств графов. В конкретных примерах алгебру данных удобно рассматривать как многоосновную алгебру.

От алгебры данных D можно легко перейти к алгебре D-функций, т. е. функций, аргументы и значения которых принадлежат D. Для этого вводится множество R-переменных и рассматриваются всевозможные отображения R в D (наборы значений переменных). Значение x функции X: $R \to D$ на переменной $r \in R$ называется значением этой переменной в наборе x. D-функция — это функция из D^x в D, зависящая только от значений переменных из некоторого конечного подмножества множества R. Множество F всех D-функций является Ω-алгеброй. Действие операций в этой алгебре определяется соотношением $(w(f_1, \ldots, f_n))(x) = w(f_1(x), \ldots, f_n(x))$. Функция f_r такая, что $f_r(x) = x(r)$ для любого $x \in D^R$ называется селекторной. Подалгебра F_0 алгебры F, порожденная селекторными функциями, является алгеброй элементарных функций. Ее можно получить также, факторизуя свободную Ω-алгебру термов $T_\Omega(R)$, порожденную множеством R по отношению к D-эквивалентности. Термы t_1 и t_2 называются D-эквивалентными $(t_1 = t_2(D))$, если для любого гомоморфизма γ: $T_\Omega(R) \to D$ выполняется $\gamma(t_1) = \gamma(t_2)$. Пары D-эквивалентных термов являются тождествами алгебры D, и естественно, что изучение конкретных алгебр данных начинается с изучения этих тождеств, используемых далее для развития техники преобразования термов (выражений в алгебре данных). Такая техника довольно хорошо освоена для числовых алгебр и реализуется во многих системах, использующих формальные преобразования. В частности, в машине «МИР-2», реализующей язык АНАЛИТИК, при выполнении аналитических преобразований широко используются тождества алгебры данных.

Большое значение имеет изучение и применение нетрадиционных алгебр данных, опыт использования которых меньше, чем опыт использования числовых алгебр. Применение соотношений алгебры данных составляет основу техники проектирования системы ПРОЕКТ. Данные этой системы представляют собой алгоритмические и структурные описания проектируемых устройств.

Операции алгебры данных можно перенести и на частично определенные D-функции, которые задаются частично упорядоченным отношением «f есть продолжение g». Если операции, распространенные на частично определенные функции, оказываются монотонными относительно упорядочения D-функций, то полученную алгебру можно расширить путем добавления функций, являющихся решениями систем уравнений вида

$$f_i = G_i(f_1, \ldots, f_n), \quad i = 1, \ldots, n, \tag{1}$$

где $G_i(f_1, \ldots, f_n)$ — выражения в алгебре элементарных функций; $f_1, \ldots, f_n$ — неизвестные функции.

Из монотонности операций следует существование наименьшего решения, которое может быть аппроксимировано последовательными приближениями $f_i^{(m+1)} = G_i(f_1^{(m)}, \ldots, f_n^{(m)})$; $f_i^{(0)}$ — нигде не определенная функция.

Уравнения вида (1) — хорошо известные рекурсивные программы, введенные Мак-Карти и интенсивно изучаемые в последнее время. Функции, которые можно определить системой (1), называются алгебраическими D-функциями.

Следующий класс моделей получается при изучении информационной среды или памяти. Фиксируем алгебру данных D и множество переменных R. Переменные из R рассматриваются как элементы памяти, а отображение $b : R \to D$ — как состояние памяти. На множестве $B = D^R$ состояний памяти рассматриваются преобразования и условия. Множество преобразований образует полугруппу, а условия являются обычными алгебраическими D-функциями, которые принимают значения 0; 1.

Соотношения в полугруппе операторов являются одним из основных источников оптимизации преобразований алгоритмов по времени работы. В простейшем случае это полугруппа операторов присваивания, т. е. операторов присваивания, имеющих вид $(r_1 := t_1, \ldots, r_n := t_n)$, где $r_1, \ldots, r_n \in R$; $t_1 \ldots, t_n \in T_\Omega(R)$.

Даже в случае свободной алгебры данных эта полугруппа имеет нетривиальные соотношения. Хотя эквивалентные преобразования выражений в полугруппе операторов присваивания изучались многими авторами, до настоящего времени хорошего алгебраического описания полугруппа операторов присваивания не имеет. В реальных системах, помимо алгебры данных, используется алгебра памяти, операции которой дают возможность описывать обращения к элементам структур данных. Оператор присваивания в этом случае имеет вид $(s_1 := t_1, \ldots, s_n := t_n)$, где $s_1, \ldots, s_n$ — выражения в алгебре памяти, $t_1, \ldots, t_n$ — выражения в алгебре данных.

Следующий уровень моделей — алгебра алгоритмов. Один из вариантов этой алгебры, разработанный в Институте кибернетики АН УССР для решения задач проектирования микропрограмм, представляет собой алгебру, состоящую из алгебры операторов и алгебры условий. Операторы определяют преобразования информационной среды, а условия являются (частичными) пропозициональными функциями, заданными на множестве состояний информационной среды. Используются следующие основные операции: полугрупповое умножение операторов, α-дизъюнкция, α-итерация, операция умножения условия на оператор, а также аналоги булевых операций для частично определенных условий. Алгебру алгоритмов, порожденную множествами элементарных операторов и условий, можно также расширить путем рассмотрения наименьших решений систем уравнений вида

$$x_i = F_i(x_1, \ldots, x_n), \quad i = 1, \ldots, n. \tag{2}$$

где x_i — операторы или условия; F_i — выражения в алгебре алгоритмов.

Для систем линейных уравнений, т. е. уравнений, правые части которых можно записать как $\bigvee_{\alpha_1}(P_1 x_1 \vee \bigvee_{\alpha_1}(P_2 x_2 \vee \ldots))$, эти решения можно представить в явном виде.

В настоящее время близкие варианты алгебры алгоритмов рассматриваются в связи с направлением, которое носит название «программирование без go to».

Динамические модели. Наиболее известными моделями алгоритмов, используемыми в прикладных областях, являются схемы программ и автоматы. Классическим примером их моделей являются машины Тьюринга. Важную роль в теории проектирования играют модели параллельных вычислений, которые в последнее время интенсивно развиваются. Все эти модели можно рассматривать с единых позиций в терминах абстрактных динамических систем.

Абстрактная динамическая система $\langle D, F\rangle$ определяется пространством D состояний и множеством F допустимых процессов функционирования системы. Элементы множества F (конечные или бесконечные) задаются последовательностью состояний. Целесообразно рассматривать также системы с непрерывным временем. В этом случае процессами являются функции времени, определенные на вещественных интервалах $[0, t)$, где $t \leqslant \infty$, и принимающие значения в множестве D. Для задач теории проектирования полезно выделить класс дискретных динамических систем, все допустимые процессы которых на любом конечном временном интервале меняют свои значения определенное число раз. Важнейшее понятие теории проектирования — понятие реализации, которое можно уточнить уже на самом абстрактном уровне. Пусть $\langle D, F\rangle$ и $\langle D', F'\rangle$ — две абстрактные динамические системы. Реализующим отображением называется отображение $\gamma : F \to F'$. Если задано реализующее отображение γ, то систему D называют реализацией системы D', а систему D' — моделью системы D. При этом если f — допустимый процесс системы D, то его образ $f' = \gamma(f)$ называется моделью процесса f, а процесс f — реализацией процесса f'.

Обычно на реализующее отображение накладывают специальные ограничения, суть которых сводится к тому, чтобы по наблюдению процесса f можно было достаточно просто определить его модель f'. Практически в ряде случаев удобно рассматривать частичные реализующие отображения $\gamma : F'' \to F'$, $F'' \subset F$, предполагая выполненным следующее условие: любой конечный допустимый процесс из F можно привести к процессу, имеющему модель.

На множестве всех конечных процессов в пространстве состояний D определено умножение: если $g : [0, t') \to D$, $h : [0, t''] \to D$, то $gh : [0, t' + t'') \to D$, $(gh)(t) = g(t)$ при $0 \leqslant t < t'$ и $(gh)(t) = h(t - t')$ при $t' \leqslant t < t''$. Время t может быть как непрерывным, так и дискретным. Множество процессов, замкнутое относительно этого умножения и содержащее пустой процесс e (единицу), называется полугруппой процессов. Полугруппа процессов G представляет собой полугруппу системы $\langle D, F\rangle$, если любой конечный допустимый процесс является началом некоторого процесса из G, а всякий бесконечный процесс из F преобразуется в «бесконечное» произведение элементов G. Отображение $\gamma : G' \to G$ полугруппы G' в G называется автоматным, если $\gamma(gh) = \gamma(g) \cdot \gamma_g(h)$. (Поскольку в полугруппе процессов выполняется закон левого сокращения, то элемент $\gamma_g(h)$ определен однозначно.)

Важнейшим классом реализующих отображений являются отображения, индуцируемые автоматными отображениями полугрупп систем.

В терминах реализующих отображений можно уточнить основные задачи теории проектирования: анализ, синтез и оптимизацию систем. В простейшем случае рассматриваются два класса систем K и K' вместе

с некоторым реализующим функтором. Этот функтор каждой системе $\langle D, F\rangle \in K$ ставит в соответствие систему $\langle D', F'\rangle \in K'$ и реализующее отображение $\gamma : F \to F'$. Задача анализа состоит в определении тех или иных свойств модели D' системы D, а задача синтеза — в построении реализации D системы D'.

Если задан некоторый критерий оптимизации, то можно говорить о выборе оптимальной реализации системы D'. Обычно на системах из классов K и K' заданы некоторые дополнительные структуры, используемые при определении реализующего функтора. Процесс проектирования состоит в построении цепочки систем, каждая из которых воплощает в себе предыдущую относительно соответствующего реализующего функтора. Классическим примером является задача проектирования конечного автомата, которая требует перехода от автоматного отображения к абстрактному автомату, затем к сети из автоматов и, наконец, к асинхронной логической сети с соответствующими средствами синхронизации. Более современный пример — последовательность все более детализированных описаний алгоритма при его проектировании сверху вниз методами структурированного программирования. Методика последовательного уточнения моделей проектируемых устройств детально разработана и широко используется в системе ПРОЕКТ.

Одна из важных формулировок задачи анализа алгоритмов состоит в определении преобразования, реализуемого данным алгоритмом, представленным с помощью некоторой динамической системы. Решение этой задачи для дискретных преобразователей можно обобщить для произвольных дискретных динамических систем. Рассмотрим это обобщение.

Поскольку дискретные динамические системы могут описывать недетерминированные вычисления, вместо алгебры преобразований удобно рассматривать алгебру бинарных отношений на множестве D. Основными операциями этой алгебры являются: обычное произведение отношений, теоретико-множественное объединение и транзитивное замыкание отношений. Пара $(\mathfrak{A}, \mathfrak{B}_0)$ алгоритмических алгебр на множестве D может быть вложена в алгебру $E(D)$ всех бинарных отношений на D, если оператор отождествить с его графиком, а условию α поставить в соответствие частичное тождественное отображение φ_α, определенное на элементах множества D, на которых α истинно. Тогда операции алгебры можно выразить следующими равенствами:

$$(P \underset{\alpha}{\vee} Q) = \varphi_\alpha P \vee \varphi_{\bar{\alpha}} Q;$$

$$\{P\} = \{\varphi_{\bar{\alpha}} P\}\, \varphi_\alpha,$$

где $\{P\}$ — транзитивное замыкание отношения P.

Операцию умножения оператора на условие нельзя выразить через основные операции алгебры отношений. Целесообразно, однако, отметить следующее важное соотношение:

$$P\varphi_\alpha = \varphi_{P\alpha} P.$$

Стандартным образом строится теория уравнений в алгебре отношений. Если $F_i(x_1, \ldots, x_n)$ — элементарные функции алгебры отношений (т. е. функции, определенные выражениями в этой алгебре, построенными из переменных и, возможно, некоторых констант), то система уравнений

$$x_i = F_i(x_1, \ldots, x_n), \quad i = 1, \ldots, n,$$

имеет наименьшее решение, которое можно представить в виде

$$S_i = \bigcup_{m=0} S_i^{(m)},$$

где

$$S_i^{(m+1)} = F_i(S_1^{(m)}, \ldots, S_n^{(m)}),\ S_i^{(0)} = \varnothing,\ i = 1, \ldots, n.$$

Системы линейных уравнений можно решать методом исключения, пользуясь формулой $X = \{S\}T$, которая выражает наименьшее решение уравнений $X = Sx \vee T$.

Пусть $\langle D, F\rangle$ — динамическая система с дискретным временем, в которой выделено множество начальных состояний D_0 и множество заключительных состояний D. Будем считать, что F состоит только из конечных процессов и вместе с каждым процессом содержит все его начальные отрезки. Рассмотрим такое отношение φ_D на множестве D, что $(\alpha, \alpha') \in \varphi \Leftrightarrow$ существует такой процесс $f \in F$, что $f(0) = \alpha$ и $f = f'\alpha'$, где $\alpha' \in D^*$.

Рассмотрим задачу нахождения отношения φ_D, представленного системой D. Решение будем искать в виде выражений в подходящей алгебре отношений. Соответствующую алгебру можно построить, если задано автоматное отображение $\gamma : F \to F(H)$ множества F допустимых процессов системы D в множестве $F(H)$ всех конечных процессов в некотором конечном множестве H. Автоматность отображения γ означает, что $\gamma(ff') = \gamma(f)\,\gamma_f(f')$, где $\gamma_f(f')$ определяется этим уравнением однозначно вследствие закона левого сокращения в полугруппе процессов. Далее будем предполагать, что длина слова $\gamma(\alpha)$ для $\alpha \in D$ не больше 1 (в случае пустого слова она равна 0) и что существуют такие множества $H_0 \in H$ и $H^* \in H$, что $\alpha \in D_0 \Leftrightarrow \gamma(\alpha) \in H_0$ и для любого f такого, что $f\alpha \in F$, выполняется $\alpha \in D^* \Leftrightarrow \gamma_f(\alpha) \in H^*$.

Рассмотрим отношения φ_h и $\varphi_{hh'}$, определенные следующим образом: $(\alpha, \alpha') \in \varphi_h \Leftrightarrow$ существует допустимый процесс вида $fdf'd' \in F$ такой, что $\gamma_f(d) = h$ и $d' \in D^*$. $(d, d') \in \varphi_{hh'} \Leftrightarrow$ существует допустимый процесс вида $fdf'd'$ такой, что $\gamma_f(d) = h$ и $\gamma_f(df'd') = hh'$, $\gamma_{fdf'}(d') = h'$.

Рассмотрим также отношение $\varphi_e = \{(d, d) \mid d \in D_0 \cap D^*\}$.

Очевидно, $\varphi_D = \bigcup_{h \in H_0} \varphi_h \cup \varphi_e$. Отношение φ_h удовлетворяет системе отношений

$$\varphi_h = \bigcup_{h' \in H} \varphi_{hh'}\varphi_{h'} \cup \bigcup_{h' \in h^*} \varphi_{h,h'},\quad h \in H.$$

Можно показать, что семейство (φ_h) $h \in H$ есть наименьшее решение системы уравнений

$$X_h = \bigcup_{h' \in H} \varphi_{hh'}X_{h'} \cup \bigcup_{h' \in H} \varphi_{hh'}.$$

Решая эту систему уравнений и подставляя полученные значения в формулу для φ_e, находим выражения отношения φ_D через элементарные отношения $\varphi_{hh'}$ в алгебре отношений.

В качестве примера можно рассмотреть задачу анализа программы A, действующей на информационном множестве B. Программа задается совокупностью переходов $\alpha \xrightarrow{\alpha/y} \alpha'$. Каждый из переходов означает, что если программа находится в состоянии $\alpha \in A$ и удовлетворяется условие α, то она может выполнить оператор y и перейти в новое состояние α'. Поставим в соответствие этой программе дискретную систему $\langle A \times B,$

F), где F состоит из всех последовательностей $\alpha_1, \ldots, \alpha_n$ таких, что $\alpha_i \to \alpha_{i+1}$, а отношение $\vdash$ непосредственного следования определяется условием $(\alpha, b) \vdash (\alpha', b')$. В программе A установлен такой переход $\alpha \xrightarrow{\alpha/\gamma} \alpha'$, что $\alpha(b) = 1$ и $\gamma(b) = b'$.

Пусть a_0 — начальное, а α^* — заключительное состояние программы. Положим $D_0 = \{(a_0, b) \mid b \in B\}$, $D^* = \{(a^*, b) \mid b \in B\}$.

Для анализа программы в качестве H можно выбрать множество A, полагая $\gamma_{f(a,b)} = a$ для любых $f \in F$. Это даст обычный алгоритм анализа. Выбирая в качестве H некоторое подмножество множества A, содержащее начальное и заключительное состояния, получаем анализ структурированной программы.

Рассматривая подобным образом части программы, расположенные между двумя состояниями из H, находим многоуровневый анализ структурированной программы (типичный технологический прием структурированного программирования).

Структурные модели. Наиболее общее понятие композиции дискретных систем определяется следующим образом. Пусть p — процесс в пространстве D, а p' — процесс в пространстве D' такой же длительности, что и p. Тогда их параллельной композицией называется процесс $q = p \otimes p$ в пространстве $D \times D'$, определенный отношением $q(t) = (p(t), p'(t))$.

Пусть F — множество процессов в пространстве D, а F' — множество процессов в пространстве D'. Тогда через $F \otimes F'$ обозначим множество всех процессов вида $p \otimes p'$, где $p \in F$ и $p' \in F'$ имеют одинаковую (конечную или бесконечную) длительность. Прямым произведением систем D и D' является система $\langle D \times D', F \otimes F' \rangle$. Система $\langle D, F \rangle$ называется подсистемой системы $\langle D', F' \rangle$, если $D \subset D'$, $F \subset F'$. Композиция систем D и D' есть любая подсистема их прямого произведения. Определение прямого произведения и композиции естественным образом обобщается на любое число систем. Если D представляет собой композицию систем $D_1, \ldots, D_n$, то эти системы — компоненты системы D.

Практические задачи инициируют изучение классов структурных моделей алгоритмов. Модели со сложной структурой естественным образом возникают при изучении параллельных вычислений.

Типичную модель современной вычислительной системы можно представить композицией следующих компонентов: памяти, блока управления, композиции алгоритмических модулей, заданных в виде автоматов или схем программ над памятью, блока ожидания, блока операционных модулей и модели внешней среды. Алгоритмические модули функционируют, изменяя свои состояния в зависимости от условий, определенных на состояниях других компонентов, и порождают операторы, которые в определенном порядке становятся в очередь в блоке ожидания. Состояние этого блока представляет собой частично упорядоченное множество мест, занятых операторами. При возникновении подходящих условий операторы передаются в блок операционных модулей, где они выполняются. В результате изменяются соответствующие компоненты состояния памяти. Состояние памяти может меняться также под действием внешней среды.

Помимо параллельного структурирования, которое описывается с помощью рассмотренной композиции, алгоритмы во многих случаях полезно структурировать сверху вниз, рассматривая последовательность систем, каждая из которых является реализацией предыдущей с помощью заданного реализующего отображения.

Метод формализованных технических заданий проектирования систем. Этот метод является одной из форм представления современной технологии алгоритмического проектирования дискретных систем. Типичными примерами дискретных систем являются программы, устройства вычислительных машин, системы программ, программно-технические комплексы, системы управления, содержащие в своем составе специализированные или универсальные ЭВМ.

Данный метод разработан в Институте кибернетики АН УССР на основе использования системы ПРОЕКТ. В этой системе он рассматривался в двух аспектах: как метод проектирования устройств и программ и как метод разработки самой системы. Метод формализованных технических заданий близок к таким направлениям, как структурированное программирование, метод последовательного уточнения программ и др. Основное отличие этого подхода от других состоит в использовании специальных математических моделей и представлении результатов на каждом этапе проектирования в виде формализованных технических заданий для последующих этапов. Метод формализованных технических заданий ориентирован как на индивидуальное решение задач проектирования, так и на коллективную работу над сложным проектом. Общие принципы метода можно сформулировать следующим образом.

1. Проектирование представляет собой многоэтапный ветвящийся процесс. На каждом этапе проектируемый объект представляется в виде набора математических моделей объекта и его частей. Эти описания составляют основу технических заданий очередного этапа. При переходе к следующему этапу осуществляется реализация моделей (движение «сверху вниз») либо построение моделей более высокого уровня (движение «снизу вверх»). В конце каждого этапа проектирования создаются технические задания для следующих этапов.

2. Обоснование свойств проектируемых объектов и установление соответствия результатов, полученных на каждом этапе, конкретным техническим заданиям осуществляются либо путем анализа математических моделей, выполняемого с помощью средств моделирования, либо путем, носящим дедуктивный характер.

3. В процессе проектирования широко используются различные способы структурирования дискретных систем, помогающие анализировать процессы функционирования дискретной системы и сводить обоснование и проектирование системы к обоснованию и проектированию ее частей.

РАЗДЕЛ 3

ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ

КИБЕРНЕТИКА И ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ

(Кибернетика и диалектика.— М.: Наука.— 1978).

О понятии «искусственный интеллект». Начальная теоретическая основа для формализации мыслительных процессов была создана, как известно, с помощью формальной логики еще в трудах Аристотеля. С тех пор задача создания искусственного разума, тесно связанная с формализацией мыслительных процессов, неизменно занимала ученых. Достаточно вспомнить знаменитую в свое время вертушку Луллия или мечты Лейбница о том времени, когда люди, вместо того чтобы спорить, будут вычислять.

Серьезная практическая постановка этой задачи стала возможной, однако, лишь в результате появления электронных вычислительных машин (ЭВМ) и развития кибернетики.

Прежде чем перейти к анализу основных принципов и путей создания искусственного интеллекта, следует более точно определить, какой смысл вкладывается в это понятие. Наиболее естественно (следуя Тьюрингу[1]) считать, что некоторое устройство (созданное человеком) представляет собой искусственный интеллект, если, ведя с ним диалог по достаточно широкому кругу вопросов, человек не сможет различить, разговаривает он с разумным живым существом (скажем, с другим человеком) или с автоматическим устройством. В способности такого устройства должны, разумеется, входить запоминание сообщенной ему информации, логический анализ, образование новых понятий и т. п. Вместе с тем не обязательно, чтобы внешняя сторона оформления диалога в полной мере соответствовала обычному человеческому разговору. Необязательны, в частности, не только наличие у рассматриваемого устройства человекоподобного образа, мимики, жестов и т. п., но даже его способность вести разговор обычным человеческим голосом и понимать обычную (звуковую) человеческую речь. Необязательно для искусственного интеллекта и наличие чего-либо напоминающего человеческое чувство зрения. Ведь не отказываем же мы человеку в признании в нем разумного существа лишь на том основании, что он глух и нем, слеп и т. п. Да и в предполагаемых контактах с возможными разумными существами (как на нашей планете, так и вне ее) мы не исходим из предположения, что внешнее оформление контакта будет соответствовать обычным человеческим стандартам.

Оставляя вопрос о расширении спектра органов чувств искусственного интеллекта на дальнейшее рассмотрение, ограничимся пока простейшей формой осуществления диалога, которую естественно назвать телеграфной. Это означает, что участники диалога должны иметь возможность обмениваться (не видя друг друга) сообщениями в той или иной (заранее

[1] См.: Тьюринг А. Может ли машина мыслить? М., 1960.

фиксированной) знаковой системе [2]. При этом подразумевается, что искусственный интеллект должен быть снабжен органами, способными воспринимать и формировать подобные сообщения, обладать памятью, позволяющей ему запоминать и хранить необходимую информацию, как вкладываемую в него заранее, так и получаемую в процессе диалога. Конечно, речь идет не о простом запоминании «слово в слово» (хотя в случае необходимости это и не исключается), а о представлении информации в такой форме, которая была бы наиболее удобной при формировании посылаемых устройством сообщений.

Совокупность правил, по которым должен работать искусственный интеллект, можно условно представить в следующей обобщенной форме. Обозначим через A информацию, содержащуюся в его памяти, а через B и C соответственно информацию в очередном получаемом и в очередном формируемом им сообщении. Тогда $C = f(A, B)$; этим выражается та очевидная мысль, что ответ C, формируемый искусственным интеллектом, определяется с помощью некоторой системы правил f из его предыдущего опыта (знаний) A и очередного полученного им сообщения B. При этом не исключается случай «пустых» сообщений B и C, т. е. умолчания при очередном вопросе или формирования очередного сообщения при отсутствии сообщения от партнера.

В случае обычного (естественного) интеллекта опыт A не остается неизменным. В процессе общения с внешним миром он пополняется и видоизменяется. Подобным свойством должен обладать, разумеется, и искусственный интеллект. Поскольку мы предположили, что его общение с внешним миром ограничивается сообщениями, то требуемое свойство интеллекта может быть выражено в виде соотношения $A' = g(A, B)$. Здесь через A' обозначен видоизмененный (после получения сообщения B) опыт (объем знаний) рассматриваемого нами искусственного интеллекта. Соотношениями $C = f(A, B)$ и $A' = g(A, B)$ полностью характеризуются поведение и закономерности развития не только искусственного, но и обычного (естественного) интеллекта. Различие между ними заключается лишь в богатстве знаковой системы (алфавита), с помощью которой происходит обмен информацией с партнером, и в природе самого партнера. В случае естественного интеллекта таким партнером является вся окружающая его действительность, а в качестве сообщений выступает весь поток арительной, звуковой и другой информации. Для искусственного интеллекта в узком смысле слова (которым мы пока ограничиваемся) вся окружающая действительность сводится ко второму участнику диалога — человеку, а в качестве знаковой системы обычно выбирается алфавит какого-либо естественного языка (латинского, русского и др.), дополненный цифрами и некоторым количеством других специальных знаков (знаки препинания, знаки математических операций и т. п.). В случае необходимости этот алфавит может быть сведен к комбинациям только двух символов (точка — тире, нуль — единица и т. п.), как это делается при передаче телеграфных сообщений.

Соотношения $C = f(A, B)$ и $A' = g(A, B)$ определяют преобразования информации, которые производятся искусственным интеллектом в процессе диалога с партнером. Нетрудно понять, что подобная ситуация наблюдается и в случае естественного интеллекта, поставленного в те же условия (режим диалога при полной изоляции от остальных внешних

---

[2] В современной теоретической кибернетике подобные знаковые системы принято называть (абстрактными) алфавитами.

воздействий). Задача создания искусственного интеллекта сводится тем самым к конструированию устройства, реализующего ту же (или достаточно близкую к ней) систему правил $C = f(A, B)$, $A' = g(A, B)$ преобразования информации, которая имеет место в случае естественного интеллекта (с соответствующими чертами индивидуальности), поставленного в условия диалога. Строго говоря, полного копирования этих правил не требуется, поскольку информация о предыдущем опыте A сама по себе в процессе диалога не наблюдаема: о ней можно судить лишь внешним образом, по формируемым сообщениям C. Поэтому важно лишь, чтобы сами сообщения C были примерно такими же, как в случае диалога с естественным интеллектом.

Теперь мы вплотную подошли к вопросу о роли ЭВМ в создании искусственного интеллекта. Современные ЭВМ способны воспринимать и формировать сообщения в упомянутом выше алфавите (буквы плюс цифры плюс знаки препинания и другие специальные знаки). Кроме того, с помощью программирования (составления и ввода в ЭВМ наборов элементарных правил преобразований информации, называемых программами) их можно настраивать на любые правила (и системы правил) преобразования информации в рассматриваемом алфавите. Подчеркнем, что речь идет о правилах любой природы, а не только о тех, которые выражаются математическими формулами. Допустимы не только правила, задающие описываемые ими преобразования информации однозначно, но и правила, при которых происходит случайный выбор требуемых преобразований. Вполне допустимым является, например, правило: «Если в очередном сообщении B партнера встретилось слово b, отсутствовавшее в предшествующем опыте A, то сформулировать с равной степенью вероятности одно из следующих двух сообщений: 1) «Вас не понял», 2) «Что означает слово b?» . В результате в одних случаях в такой же ситуации искусственный интеллект дает разные ответы даже при неизменном опыте A. Изменение же этого опыта (например, разъяснение партнером смысла слова b) может привести к полному изменению ответа.

Разумеется, приведенный пример, когда возможные ответы в правиле заранее перечислены, далеко не типичен. Обычно в правилах определяется метод конструирования ответа, приводящий (в зависимости от опыта A и вопроса B) к неограниченно большому множеству возможных ответов.

Возможность программирования в ЭВМ любых правил преобразования информации вытекает из алгоритмической полноты системы их элементарных операций. Факт наличия такой полноты установлен здесь со степенью достоверности во всяком случае не меньшей, чем достоверность любого из известных сегодня фундаментальных законов природы. В то же время одним из краеугольных камней, на котором зиждется последовательно материалистическая философия, является принцип познаваемости любых явлений, в частности и правил преобразования информации $C = f(A, B)$ и $A' = g(A, B)$ в любом естественном интеллекте. Но, будучи познанными, эти правила могут быть запрограммированы (причем уже в современных ЭВМ). Снабженная соответствующей программой (точнее, системой программ) ЭВМ и будет представлять собой искусственный интеллект в рассматриваемом нами (узком) смысле.

Целесообразно остановиться теперь на некоторых заблуждениях, связанных с программированием искусственного интеллекта. В основе первого из них лежит наивное представление об ЭВМ, как об устройстве, реализующем лишь строгие математические процедуры, выбирающем всегда и во всех случаях лишь самые лучшие решения и т. п. Дело доходило

до сравнений искусственного интеллекта ЭВМ с пресловутым буриданов ым ослом.

Разумеется, в ряде случаев вследствие огромного быстродействия ЭВМ на них удается реализовать сложные математические процедуры, недоступные «невооруженному» человеческому мозгу. Однако ситуация буриданова осла в работе ЭВМ не может возникнуть разве что в результате грубой ошибки составителя программы. Большинство же задач, возникающих перед искусственным интеллектом сегодня и в обозримом будущем, вообще недоступно точным математическим методам. Программируя эти задачи на ЭВМ, мы ставим себе целью найти не наилучшее, а лишь одно из достаточно хороших решений. При решении таких задач в программах воплощаются по сути те же (с поправкой на быстродействие ЭВМ) методы поиска решений, которые используются человеком. Этот весьма очевидный прием использовался фактически с самого начала развития ЭВМ и получил затем специальное название: эвристическое программирование.

На принципе эвристического программирования строятся сегодня по сути все сколько-нибудь интересные программы, реализующие те или иные стороны искусственного интеллекта.

Второе заблуждение имеет несколько более тонкую природу: допуская, что в принципе на ЭВМ можно реализовать любые правила переработки информации, вместе с тем утверждают, что у человека эти правила не остаются неизменными, они дополняются и совершенствуются по мере накопления знаний и опыта, тогда как искусственный интеллект представляет собой нечто застывшее, в противоположность человеческому разуму, который непрерывно развивается.

Ошибка в приведенном рассуждении заключается в игнорировании того факта, что системы правил (программы) в ЭВМ записываются в память точно так же, как и любая другая информация. Поэтому они могут быть подвергнуты любым изменениям по мере накопления опыта. Для этого изучаются и программируются правила изменения правил. Данные правила второй ступени, в свою очередь, могут быть подвергнуты любым изменениям с помощью правил третьей ступени и т. д. Иными словами, на современных ЭВМ в принципе может программироваться поведение сколь угодно сложных самообучающихся и самосовершенствующихся систем [3]. При этом в качестве правил верхней ступени, управляющих процессом самосовершенствования, могут использоваться лишь общие законы эволюции органического мира.

Третье заблуждение (которое разделяли некоторые крупные ученые) основано на убеждении в том, что искусственный интеллект способен лишь решать поставленные ему задачи, а постановка новых задач находится вне пределов его компетенции. Ясно, однако, что правила, которыми руководствуется человеческий мозг при постановке новых задач, принципиально точно так же познаваемы, как и любые другие правила, а если они могут быть познаны, то могут быть и запрограммированы. Более того, в некоторых узких областях деятельности правила постановки новых задач могут оказаться даже проще, чем правила их решения (хотя в целом, по-видимому, имеет место обратное). Во всяком случае в предполагаемом диалоге искусственного интеллекта с человеком обе стороны должны уметь не только отвечать, но и задавать вопросы.

[3] Возникающие на практике ограничения связаны не с принципиальными, а с чисто техническими причинами, такими, например, как недостаточный объем памяти, трудности составления больших программ и т. п.

Четвертое заблуждение состоит в утверждении, что искусственный интеллект будет навсегда лишен форм проявления человеческих эмоций и черт индивидуальности характера. Разумеется, практические потребности сегодняшнего дня требуют изучения и программирования прежде всего правил рассудочной деятельности. Однако в случае необходимости не представляет особого труда наделить соответствующие системы программ формами проявления эмоций и другими индивидуальными «чертами характера», проявляющимися в формируемых ЭВМ сообщениях. Практика показала, что эта задача проще, чем задача программирования рассудочной деятельности в сколько-нибудь широкой области.

Пятое заблуждение (бытующее в научной фантастике) состоит в том, что решение задачи создания искусственного интеллекта сводится к гениальной догадке, открытию чего-то вроде «формулы мышления». При этом забывают, что человеческий интеллект представляет собой одно из самых сложных, если не самое сложное, созданий природы. Поэтому и задача создания искусственного интеллекта (а не отдельных его элементов) является сложнейшей научно-технической задачей из всех, которые когда-либо решало человечество. Решение этой задачи в сколько-нибудь полной мере возможно лишь в результате длительных организационных усилий больших научных и конструкторских коллективов, может быть, не одного поколения ученых.

Практические пути создания искусственного интеллекта. Установив факт принципиальной возможности создания искусственного интеллекта на базе современных ЭВМ, перейдем к описанию практических путей подхода к решению этой задачи.

Существует несколько различных путей, ведущих к созданию искусственного интеллекта. Первый состоит в накоплении программ, автоматизирующих различные области интеллектуальной деятельности и объединяемых по мере их создания в единую систему. Чтобы оценить сложность подобной задачи, достаточно отметить, что создание только лишь комплекса программ (с необходимой информационной базой), позволяющей ЭВМ играть в шахматы на уровне гроссмейстера, потребует не одного десятка человеко-лет работы опытных программистов и аналитиков шахматной игры. Ведь сегодня можно с достаточной уверенностью утверждать, что попытки автоматизации шахматной игры на уровне гроссмейстера с помощью одной программы (точнее, с помощью программы, основанной на едином принципе) бесперспективны. Необходимо, по крайней мере, ввести в ЭВМ все современные знания о дебютах и большое количество различных программ для разыгрывания эндшпилей разных видов. Но автоматизация шахматной игры — только одна небольшая часть задачи создания искусственного интеллекта. Что же говорить о всей задаче в целом? Ясно, что при современном уровне развития ЭВМ и техники программирования количество труда, необходимого для ее решения, выразилось бы огромными цифрами.

Не следует, однако, забывать, что путь простого накопления программ не единственно возможный. Он может и должен быть органически объединен с задачей совершенствования ЭВМ и техники программирования. Необходимость решения этой задачи обусловливается тем, что современные ЭВМ реализуют непосредственно (с помощью соответствующих схемных решений) лишь простейшие преобразования информации (запись, чтение, сравнение слов и чисел, сложение чисел и т. п.), называемые машинными командами (инструкциями). Хотя из этих «элементарных кирпичиков» в принципе можно сложить любые, сколь угодно сложные системы правил

преобразования информации, на практике (особенно в случае сложных задач) они оказываются слишком мелкими. В результате машинные программы (т. е. представление правил в виде последовательности машинных инструкций) оказываются весьма громоздкими, а следовательно, и весьма трудоемкими. Даже простейшие задачи невычислительного характера, которые должен решать искусственный интеллект, содержат, как правило, многие тысячи машинных инструкций.

Для задач вычислительного характера дело обстоит несколько лучше, поскольку с самого начала системы команд ЭВМ ориентировались именно на этот класс задач. Что же касается задач невычислительного характера (включая аналитические преобразования в математике), то для их программирования исходный «интеллект» большинства современных ЭВМ (воплощенный в их системе команд) оказывается практически весьма малоэффективным.

Основная идея совершенствования ЭВМ в этом плане состоит в повышении их исходного интеллекта, т. е. в схемной реализации гораздо более богатого набора «крупноблочных» машинных операций. Это направление (впервые в мире) было реализовано в серии отечественных ЭВМ «МИР». Уже в машине «МИР-1» (1965 г.) был реализован богатый набор операций для работы с целыми числами любой разрядности, рядами, определенными интегралами и т. д., которые в обычных ЭВМ требуют достаточно сложного программирования. В ЭВМ «МИР-2» (1969 г.) этот набор расширен на задачи аналитических преобразований. В результате исходный машинный интеллект в ряде вопросов повысился до уровня знаний студента второго курса. В последующих моделях ЭВМ типа «МИР» в исходный интеллект машины будет включена способность к логическим рассуждениям на уровне того, что современные математики считают возможным выражать словами: «Отсюда с помощью очевидных умозаключений легко доказать, что...».

Последовательность шагов в развитии исходного машинного интеллекта, принятая в машинах серии «МИР», определяется, с одной стороны, технологическими возможностями электронной промышленности, а с другой — практическими потребностями в ЭВМ определенного класса. Тем самым удается реализовать принцип единства дальних и ближних целей, т. е. задачи создания искусственного интеллекта и непосредственные практические задачи сегодняшнего дня в машинах с упрощенным программированием для широкого использования в НИИ и КБ.

Развитие техники программирования также идет прежде всего по линии создания набора крупных блоков, так называемых стандартных программ, для последующего их использования при программировании различных задач. Кроме того, для программирования используются языки, более приближенные к обычным естественным языкам, а процесс перевода записанных на них программ в машинные программы берут на себя специальные сложные программы-переводчики (трансляторы или интерпретаторы). Эффективное использование программами различных машинных ресурсов (например, различных видов памяти), требующее особых усилий от программистов, осуществляется сегодня специальными организующими системами программ — так называемыми операционными системами.

Наиболее эффективным является соединение описанных приемов развития техники программирования с непрерывным совершенствованием исходного машинного интеллекта. При этом важно, чтобы при совершенствовании систем машинных команд сохранялась преемственность, поз-

воляющая использовать на новых ЭВМ программный задел, созданный для ЭВМ предшествующих поколений. Только при этом условии описанный путь постепенного накопления программ, автоматизирующих различные виды интеллектуальной деятельности, приведет к конечной цели — созданию на их базе искусственного интеллекта, действительно оправдывающего это название.

К настоящему времени в мире накоплен уже довольно большой набор программ, реализующих (или, по крайней мере, предвосхищающих) многие стороны будущего искусственного интеллекта. Здесь прежде всего следует назвать многочисленные программы для игры в шашки, шахматы и другие игры, программы перевода с одного языка на другой, программы для доказательства теорем (пока эффективные главным образом в начальных разделах математики), программы для автоматизации некоторых процессов музыкального, литературного и художественного творчества. Существуют программы, моделирующие «личность» с определенным (пока еще весьма небольшим) набором знаний в некоторой узкой области, успешно реализуется диалог с такими «личностями».

Принципиально важное значение имеет создание системы программ, позволяющей ЭВМ «овладеть» тем или иным естественным человеческим языком. В этом состоит одно из важных направлений развития автоматизации программирования и исходного интеллекта ЭВМ, целью которого является добиться программирования ЭВМ на естественном языке подобно тому, как это делается при выдаче различных заданий человеку.

После достижения этой цели задача дальнейшего формирования искусственного интеллекта перейдет от кибернетиков к педагогам, психологам, специалистам различных областей знания.

Задача полного овладения естественным человеческим языком предполагает определенные элементы самоорганизации и самосовершенствования соответствующей системы программ в ЭВМ. Необходимо, в частности, иметь возможность вводить новые слова и понятия, объясняя их значение ЭВМ с помощью уже известных ей языковых средств точно так же, как это делается при обучении человека. В последние годы подобные расширяемые языковые системы получили развитие в рамках формальных языков программирования (например, АЛГОЛ-68).

Последовательное проведение принципа самоорганизации приводит к еще одному возможному пути построения искусственного интеллекта. Речь идет о программировании таких минимальных принципов самоорганизации, которые были бы достаточными для возникновения и последующего совершенствования искусственного интеллекта в результате процесса обучения (либо с помощью человека, либо даже в результате непосредственного взаимодействия с внешним миром).

В связи с этим следует предупредить еще об одном заблуждении. Совсем недавно был широко распространен взгляд, согласно которому в результате взаимодействия с внешней средой (лишенной интеллекта) искусственный интеллект может быть построен существенно более просто и быстро (по сравнению с описанным выше способом). Наиболее крайние приверженцы этого взгляда выражали уверенность, что такого результата можно достигнуть даже при нулевой начальной организации искусственного интеллекта, т. е. при полной случайности начальных связей между элементами, из которых собирается устройство, его реализующее.

Разумеется, в принципе такой путь построения искусственного интеллекта возможен. Однако для его реализации потребовалось бы, во-пер-

вых, запрограммировать все закономерности, управляющие процессом эволюции органического мира, и, во-вторых (и это самое трудное), фактически повторить в информационных воздействиях на создаваемый искусственный интеллект всю историю эволюции от первичных белковых соединений до человеческого мозга. Разумеется, возможны и другие варианты биологической эволюции, однако можно почти наверное утверждать, что они не будут существенно проще варианта, имевшего место на Земле.

Сегодня представляется достаточно ясным, что практически наиболее быстрым и эффективным путем создания искусственного интеллекта является разумная комбинация двух описанных нами путей. Иными словами, необходимо сообщить создаваемому искусственному интеллекту достаточно высокий начальный уровень организации, а для дальнейшего его совершенствования использовать принципы обучения и самоорганизации.

Отметим, что, подчеркивая трудность задачи создания искусственного интеллекта, мы имеем в виду интеллект, во всех отношениях равный или даже превосходящий человеческий. Дело в том, что относительно нетрудно (и это фактически уже сделано сегодня) создать систему программ, в процессе диалога с которой поверхностный наблюдатель может прийти к выводу, что он имеет дело с настоящим разумом. Однако попробуйте перевести диалог на другую тему и сделать его более длительным (например, предпринять попытку обучить партнера по диалогу какой-либо новой игре, неизвестному для него языку и т. п.), и вы сразу убедитесь в узости и ограниченности способностей такого «разума».

Говоря об искусственном интеллекте и его подобии естественному человеческому интеллекту, мы все время имели в виду подобие не самих мыслительных аппаратов, а лишь их внешних проявлений (в частности, в диалоге). Иными словами, речь шла о феноменологическом подходе к моделированию мыслительных процессов и созданию искусственного разума. Определенный интерес представляет еще один подход к проблеме создания искусственного интеллекта, когда объектом моделирования является мыслительный аппарат человека — его головной мозг. Этот подход, в свою очередь, распадается на два различных направления. Первое из них основано на создании системы программ для ЭВМ, имитирующих поведение как отдельных нейронов, так и их взаимодействие. Второе направление использует физическое моделирование нейронов и фактическое их объединение в сложные структуры по тем же принципам, по которым это сделано в мозгу.

Подход, основанный на моделировании мозга, на первый взгляд представляется весьма перспективным. Прежде всего привлекает то, что объект моделирования состоит из однотипных элементов — нейронов. Не следует, однако, забывать, что этих элементов в мозгу очень много (не менее десяти миллиардов). В то же время нынешний уровень развития электроники позволяет в лучшем случае соединить в одном устройстве несколько миллионов логических элементов, которые к тому же заведомо более просты, чем нейроны. Не следует также забывать, что основная сложность строения мозга заключена не в самих нейронах, а в способе их соединения друг с другом. Число точек соединения нейронов — синапсов на два порядка выше, чем число самих нейронов. Именно эти точки воплощают в себе долговременную память мозга, превращающую его из простого скопища нейронов в эффективный мыслительный аппарат. Правда, синоптические контакты «воспитуемы». Их проводимости, а значит, и конфигурация связей нейронов изменяются в процессе работы мозга. Несомненно,

что значительная доля этих связей устанавливается уже после рождения человека в процессе его воспитания и обучения.

Однако столь же несомненно, что весьма большой объем информации о начальной структуре связей нейронов передается генетическим путем. Поэтому неосновательны надежды без особых трудностей воссоздать мыслительный аппарат человека из небольшой (а тем более нулевой) начальной организации, т. е. из системы случайно связанных друг с другом нейронов.

Проникнуть же в тайны начальной организации мозга (не говоря уже о структуре связей нейронов на последующих стадиях его развития) не вероятно трудно. Это связано с тем, что проводимость синаптических контактов пока недоступна прямому экспериментальному определению. К тому же не так просто установить на живом мозге, к какой паре нейронов относится наблюдаемый синаптический контакт. Судить о структуре связей нейронов и проводимости соответствующих синаптических контактов в настоящее время можно лишь косвенным образом — в результате измерения электрической активности групп нейронов. Эксперименты, ставящиеся с этой целью, весьма тонки и достаточно трудоемки. Обработка их результатов (с точки зрения задачи создания искусственного интеллекта) также представляет собой нелегкую задачу. Поэтому сегодня можно с достаточной степенью уверенности утверждать, что разгадка начальной организации и других тайн мозга системного характера вряд ли явится результатом прямых физиологических экспериментов. Больших результатов сегодня можно добиться при проверке тех или иных гипотез о структуре мозга на математических или физических моделях. Разумеется, такие гипотезы следует строить с учетом накопленного экспериментального физиологического материала и анатомических данных, а проверку на моделях дополнять прямой экспериментальной проверкой.

Таким образом, основное значение прямого моделирования мозга в современных условиях заключается именно в разгадке тайн его работы. Что же касается практической стороны задачи создания искусственного интеллекта, то прямое моделирование мозга следует признать в настоящее время достаточно бесперспективным. Дело здесь не только в огромном количестве нейронов и сложности организации их взаимосвязей. Не менее важно то, что принципы организации сложных систем в электронике на современном уровне развития электронной технологии принципиально отличны (и не могут не быть отличными) от принципов, по которым строятся биологические системы. Поскольку каждый электронный элемент имеет определенную стоимость, одним из основных принципов построения систем в электронике является принцип экономии элементов. В то же время природа может позволить себе быть расточительной. Это как раз и имеет место в случае мозга, где практически все важные функции, определяющие его интеллект (включая долговременную память), «размазаны» по большей части составляющих его элементов [4]. Сегодня уже достаточно ясно, что такая расточительность (с чисто системотехнической точки зрения

[4] Высказанное утверждение на первый взгляд противоречит давно установленному факту о локализации различных функций на определенных участках коры головного мозга. Естественно, однако, предположить, что на этих участках сосредоточены лишь нейроны, непосредственно связанные с определенными органами чувств или мышцами. Они представляют собой, таким образом, своеобразные устройства ввода и вывода информации, осуществляющие одновременно некоторые ее простейшие преобразования. Что же касается основных способностей собственно мыслительного характера, то, по всей видимости, они рассредоточены по всем остальным нейронам коры одновременно.

является излишней. Это означает, что при рациональном подходе к конструированию мыслительного аппарата (из обычных биологических нейронов) те же функции и, что самое существенное, ту же (и даже большую) надежность можно было бы обеспечить с помощью значительно меньшего числа нейронов, чем это происходит в живом мозгу.

Принципы системной организации, заложенные в структуре мозга, можно будет в полной мере использовать при конструировании электронных устройств лишь при одном условии, а именно должна быть разработана технология, в которой производство нейроноподобных элементов практически ничего не стоит, а установление случайных связей между ними (точнее, связей с большим элементом случайности) намного проще, чем установление строго детерминированных связей. Хотя возможность создания такой технологии не исключена, надежда на то, что это произойдет в обозримом будущем, весьма невелика. Поэтому наиболее вероятно, что задача создания искусственного интеллекта будет решена на описанных выше феноменологических принципах, а не в результате непосредственного моделирования мозга, как системы нейронов.

Искусственный интеллект и роботы. До сих пор мы вели разговор лишь об искусственном интеллекте в узком смысле слова, не ставя целью придать ему человекообразный вид. Мы, в частности, оставляли пока в стороне вопрос о наделении искусственного интеллекта зрением и другими органами чувств. Существует, однако, немало практически важных задач автоматизации (прежде всего автоматизации технологических процессов в машиностроении, строительстве, транспорте и др.), требующих решения именно этих вопросов. Речь идет о создании универсальных промышленных роботов, которые могли бы выполнять (и притом гораздо лучше, чем человек) любую работу на производстве, в настоящее время выполняемую вручную. Универсальность роботов требует наличия у них достаточно развитого интеллекта, что может быть достигнуто лишь при использовании достаточно мощных ЭВМ. Вместе с тем эти роботы должны иметь способность свободно перемещаться, иметь некоторый аналог человеческих рук, а также обладать по крайней мере двумя человеческими чувствами — зрением и осязанием [5]. Для упрощения общения с роботом желательно также иметь возможность разговаривать с ним обычным человеческим голосом.

Задача воссоздания человеческих органов чувств не ограничивается одними лишь датчиками (для органов зрения, слуха и осязания такие датчики давно созданы). Гораздо более трудной является задача обработки информации, поступающей от датчиков, с целью ее последующей передачи в более высокие звенья интеллекта. Смысл этой задачи заключается в так называемом распознавании образов. Дело в том, что одна и та же сущность может быть выражена в различных формах, например слово, произнесенное разными голосами, картинки с изображениями различных домов или даже одного и того же дома, изображенного в различных ракурсах, и масштабах, и т. п. Устройство, осуществляющее распознавание, должно уметь выделять одну и ту же сущность, выраженную в различных формах. У человека такая задача решается специальными «вводными» системами нейронов. Для зрения, передающего в мозг особо большие объемы информации, такая система включает в себя чуть ли не половину ней-

---

[5] Последнее необходимо, чтобы соразмерить усилия, вкладываемые в «руки» робота, с оказываемым им сопротивлением, т. е. весом, твердостью и другими аналогичными свойствами предметов, с которыми ему приходится иметь дело.

ронов, имеющихся в головном мозгу. Это показывает, что задачи распознавания образов, особенно зрительных, являются чрезвычайно сложными. Трудность их решения состоит еще и в том, что процессы, происходящие в соответствующих отделах мозга, почти не контролируются сознанием даже в чисто феноменологическом плане подобно тому, как это происходит в случае логического мышления. Кроме того, в мозгу решение задач распознавания убыстряется за счет параллельной работы сотен миллионов и даже (в случае зрения) миллиардов нейронов. В современных же ЭВМ их приходится расчленять на совокупность шагов, выполняемых последовательно, один за другим. В результате сводится на нет то огромное преимущество в быстродействии элементов (по сравнению с биологическими элементами — нейронами), которым обладают ЭВМ.

Хотя решение задач распознавания образов находится еще в самой начальной стадии, сделано уже немало. Созданы устройства (ЭВМ с соответствующими датчиками и программами), способные распознавать простые геометрические тела (кубы, цилиндры, пирамиды и т. п.), представленные в различных ракурсах, по форме, цвету и размерам. Решена задача распознавания речи с ограниченным словарным запасом (порядка сотни слов при произнесении их различными голосами и порядка тысячи — при одном голосе). Задачу вывода информации синтетическим машинным голосом (говорящие ЭВМ) фактически можно считать уже решенной.

Отметим, что задачи распознавания образов возникают не только в «периферийных» отделах мозга, но и в процессе логического мышления. Ведь естественные человеческие языки предоставляют возможность выражения одной и той же мысли в самых различных языковых формах. В процессе мышления мозг производит перевод мыслей, выражаемых средствами обычного «внешнего» языка, в некоторый (пока еще недостаточно ясно представляемый) «внутренний» язык смысловых понятий. На этот же язык переводится, по всей видимости, и информация, поступающая в мозг из других органов чувств.

В распознавании «языковых» образов достигнуты определенные успехи, однако в целом эта задача еще далека от окончательного практического решения.

Успехи в решении задач распознавания образов и автоматизации некоторых сторон «рассудочной» деятельности позволили создать первые универсальные промышленные роботы, способные работать на разнообразных подъемно-транспортных и простых сборочных операциях. Препятствием к широкому распространению таких роботов является еще очень высокая их цена. Поэтому в ближайшем будущем универсальные роботы будут применяться прежде всего для работы в трудных и опасных условиях (под водой, в космосе, при наличии опасных излучений и т. п.). Но следует забывать, что роботы отнюдь не являются чем-то вроде ухудшенной копии человека. При условии решения соответствующих задач распознавания образов им нетрудно придать органы чувств, отсутствующие у человека, например способность «видеть» электрические и магнитные поля, рентгеновское и гамма-излучение и т. п. Искусственным «рукам» роботов можно придать любую силу, их снабжают большим числом степеней свободы по сравнению с руками человека, что позволяет им производить работы в труднодоступных местах (например, внутри сильно изогнутых труб).

Дальнейшие успехи в совершенствовании роботов (прежде всего их «органов чувств»), а также в расширении сферы их использования в зна

чительной мере будут определяться успехами электронной технологии. Дело в том, что (как уже отмечалось выше) задачи распознавания образов более естественно и эффективно решаются с помощью специальных схемных (а не программных) решений. Такие схемы должны включать в себя огромное количество логических элементов с соответствующим образом организованными связями между ними *. Технологическая база ЭВМ первого и второго поколений и даже нынешнего, третьего поколения не позволяла по-настоящему решить эту задачу, поскольку сложные схемы в них приходилось собирать, по сути, из отдельных элементов. Правда, в ЭВМ третьего поколения используются так называемые интегральные схемы, которые представляют собой несколько логических элементов, создаваемых на одном миниатюрном кусочке полупроводника в результате единого технологического процесса. Однако степень интеграции (т. е. число логических элементов в одной интегрированной схеме) в элементах третьего поколения очень мала (4—10 логических элементов на схему). Появление и развитие больших интегральных схем (БИСов) привели к повышению степени интеграции до нескольких тысяч логических элементов на схему. В результате возникли реальные перспективы создания сложных схем (разумной стоимости и достаточной надежности), необходимых для эффективного решения задач распознавания образов.

Открывающиеся технологией БИСов перспективы резкого снижения стоимости ЭВМ и дальнейшего улучшения их параметров (скорости, объема памяти и др.) создают возможность более широкого распространения роботов и других нетрадиционных применений ЭВМ. Прежде всего это относится к различным аспектам проблемы создания искусственного интеллекта.

Отметим, что снабжение искусственного интеллекта развитым набором «органов чувств» и устройств, позволяющих ему активно влиять на окружающую среду (прежде всего искусственных «рук» и средств передвижения), имеет не только чисто практическое, но и глубокое принципиальное значение. Ведь рассматривавшийся выше искусственный интеллект был, так сказать, интеллектом в чистом виде, предназначавшимся лишь для абстрактного мышления. Единственным средством общения с окружающей средой был человек — его партнер по диалогу. Иное дело искусственный интеллект, соединенный с возможностями робота. Это уже интеллект в широком смысле слова, способный, подобно человеческому, реализовать все три составные части известной ленинской гносеологической формулы: «От наглядного созерцания к абстрактному мышлению и от него к практике».

Искусственный интеллект в широком смысле слова тем самым оказывается способным к самостоятельному дальнейшему исследованию, активному эксперименту, открытию новых законов природы и т. п. Разумеется, в чисто практическом плане это еще достаточно отдаленная перспектива. Ясно, однако, что подобный уровень развития техники ставит перед обществом серьезные проблемы, думать о которых следует уже сегодня.

В заключение отметим, что работы по созданию искусственного интеллекта и даже отдельных его функциональных звеньев имеют огромное научное и прикладное значение. Как уже говорилось выше, эти работы позволяют приблизиться к разгадке одной из самых сокровенных тайн природы — полному пониманию того, что такое мышление. Несомненно,

---

* Для определения характера таких связей значительную пользу могут принести и уже приносят успехи, достигнутые в экспериментальной нейрофизиологии.

это поможет решению многих практических вопросов, интересующих медицину, психологию и педагогику.

Еще более важное практическое значение имеет другой аспект проблемы создания искусственного интеллекта. Дело в том, что на пути совершенствования естественного человеческого интеллекта встречаются принципиальные ограничения чисто биологического характера (ограниченность числа нейронов, их относительно небольшое быстродействие и, наконец, ограниченный срок жизни человека). Разумеется, еще далеко не все резервы человеческого мозга исчерпаны. Многое может быть сделано на пути совершенствования обучения и воспитания (особенно в самом раннем возрасте). Возможно, скажет еще свое слово и медицина. Принципиально важное значение имеет то, что человечество научилось преодолевать ограниченность возможностей отдельного человеческого мозга путем общественного разделения труда (в данном случае умственного). Тем не менее сам факт существования ограничений биологического характера не подлежит сомнению. В то же время для искусственного интеллекта подобные ограничения — если они в принципе и имеют место — находятся так далеко, что практически сегодня еще не просматриваются. Поэтому принципиально нет помех для создания искусственного интеллекта, значительно превосходящего естественный человеческий интеллект. Нетрудно понять, какие возможности открываются тем самым прежде всего для развития науки и техники. Создавая отдельные функциональные звенья будущего искусственного интеллекта, ученые уже сейчас стремятся к тому, чтобы они существенно превосходили соответствующие возможности человеческого интеллекта. Этого уже удалось добиться в ряде простых игр, в решении многих проектно-конструкторских задач творческого характера, не говоря уже о чисто вычислительных задачах. Но в целом уровень автоматизации пока таков, что в одних компонентах, определяющих успех деятельности в соответствующей области, преимущество пока сохраняется за естественным (обычно незаурядным) интеллектом, хотя в других оно уже перешло к искусственному интеллекту.

Поэтому в практическом отношении при решении сложных задач творческого характера сегодня и в ближайшем будущем наиболее перспективны смешанные (человеко-машинные) системы. Такие системы должны соединять в себе наиболее сильные стороны как человеческого, так и машинного интеллекта.

В связи со всем сказанным уместно отметить, что нет оснований опасаться за судьбы человеческого интеллекта в условиях быстрого прогресса в создании искусственного интеллекта, хотя такие опасения и существуют.

(НТР и развитие художественного творчества.—
Л.: Наука, 1980)

Проблема использования кибернетики в различных областях науки и культуры, т. е. в различных областях творческого труда, была и остается самой интересной и перспективной. Эта проблема рассматривалась на многих конференциях и симпозиумах у нас и за рубежом, и количество посвященных ей работ растет с каждым днем. Лет десять назад споры вокруг темы «Кибернетика и творчество» носили характер во многом чисто теоретический. Теперь пришло время реалистического обсуждения современного положения дел и дальнейших поисков.

Вначале несколько слов о возникающих трудностях.

На данном этапе развития кибернетики еще нельзя считать окончательно сложившейся терминологию этой науки. Некоторые применяемые термины, такие, например, как «общение человека с машиной», «искусственный интеллект» и другие, в известной мере условны, метафоричны, хотя уже вошли в специальную литературу (отметим, что проблема «искусственного интеллекта» именно в этом обозначении включена в план работы Комитета по системному анализу при Президиуме Академии наук СССР). Такие выражения, как «автоматизация творческого процесса», часто режут слух представителям литературоведческих и искусствоведческих специальностей, более того, в них видят грубое посягательство на тончайшие и сложнейшие области духовной жизни человека [1]. Но здесь необходимо подчеркнуть, что сам термин «автоматизация» в данном случае не имеет обыденного, привычного смысла. Даже в тех ограниченных пределах, в которых уже применяется или будет применяться ЭВМ в определенных областях творчества, не может быть и речи (тем более сегодня) о передаче всех полномочий машине.

В связи со сказанным остановимся на вопросе о сути общения человека с машиной. Это общение упрощено за счет широкого развития различных устройств ввода и вывода, которые помогают нам давать ЭВМ информацию и получать информацию от нее в привычной для нас форме. Уже созданы различные читающие автоматы, способные воспринимать печатный текст, и ставится вопрос о вводе в машину заданий непосредственно с голоса. Вопрос ввода в ЭВМ текста непосредственно с машинописных лис-

[1] Как положительное явление отметим факт включения проблемы «Кибернетика и творчество» в круг проблем, интересующих Комиссию комплексного изучения художественного творчества при Научном совете по истории мировой культуры АН СССР. Так, в выпущенном ею сборнике «Художественное и научное творчество» (Л., 1972) целый раздел посвящен рассмотрению этой проблемы на различных уровнях (как философско-теоретическом, так и практическом). Здесь обсуждаются и серьезные практические вопросы использования ЭВМ в целях рационализации труда в литературоведении и искусствознании. Перспективна и сама установка на объединение усилий специалистов-кибернетиков и специалистов в области изучения художественной культуры.

тов, можно сказать, уже решен теоретически. Однако трудности экономического характера сдерживают возможность практического его решения.

Общение человека с машиной упростится, если ученые и конструкторы до максимума повысят «интеллектуальность» ЭВМ, т. е. заложат в нее определенную способность «мышления», чтобы не «объяснять» ей элементарных вещей (например, как найти синус какой-то величины, и т. п.).

Надежная техническая база, которая совершенствуется из года в год, дает возможность реально поставить вопрос об автоматизации определенных процессов, например, в таких сферах человеческой деятельности, как живопись, музыка. Первые попытки в этом направлении дали хорошие результаты. Программы, созданные для ЭВМ, обнаруживали достаточный уровень «интеллектуальности».

Каковы же конкретные возможности и результаты применения ЭВМ в различных областях творчества в настоящее время?

Начнем с примеров, касающихся архитектуры и конструкторского труда. Если говорить об автоматизации конструкторского труда (с помощью разных систем автоматизации проектирования), то здесь роль человека сохраняет свое важнейшее значение. Скажем, проектируются жилые дома или пассажирские теплоходы. Машина может намного лучше, чем человек, разместить каюты в корабле или жилые комнаты в доме, но может и допустить элементарную ошибку (где-то не будут открываться двери в коридор), поскольку в программе не была предусмотрена эта «мелочь». А конструктор сразу видит все недостатки проекта, быстро оценивает их и устраняет.

В Институте кибернетики АН УССР уже создана автоматизированная система для проектирования новых вычислительных машин. Состоит она не из каких-то специальных агрегатов и установок, а из набора программ. Все задания по проектированию машина решает под контролем человека, ведущего с ней диалог.

Конечно, было бы намного проще задать машине лишь исходное задание: спроектировать ЭВМ с определенной скоростью, определенного назначения и стоимости. Однако машина пока еще не способна выполнить такой приказ. Она даст лишь структурную схему задания, в которой будут указаны основные блоки будущей конструкции и характер их взаимодействия, больше ничего без нашего приказа электронный мозг сделать не в состоянии. После этого мы поручаем ЭВМ вычислить структуру каждого блока. Когда и эта работа выполнена успешно, приказываем перейти к последней стадии проектирования — созданию чертежей и схем. Таким образом, весь процесс идет по линии все более детализации и углубления, так сказать, в недра будущей ЭВМ, и все это происходит при постоянном диалоге машины с человеком. Да иначе и быть не может. Ведь случается, что машина предлагает технически неосуществимые проекты.

Новый метод значительно облегчает работу конструктора. Если раньше на конструирование одной большой машины многотысячный коллектив затрачивал более пяти лет, то сегодня двадцать человек с помощью электронного мозга выполняют эту работу за месяц.

В автоматизированных системах учтена обратная связь конструктора с машиной. План квартиры, отраженный на экране, дает возможность конструктору оценить расположение комнат. Он считает, например, что какую-то стену нужно переместить вправо или влево, и тут же сообщает об этом машине. С помощью светового карандаша конструктор перечеркивает «дефектную» деталь и стрелкой показывает, в каком направлении и на какое расстояние следует передвинуть стену. Перед ним сразу же возникает

новый чертеж, который снова оценивается. Так продолжается до тех пор, пока конструктор не решит окончательно, что чертеж полностью соответствует его требованиям. Таким образом, человек все время направляет поиск. Машина лишь предлагает варианты, но последнее слово остается за человеком. Желательно, конечно, чтобы машина сама могла отбрасывать бесперспективные варианты проекта. Но для этого в нее нужно заложить определенную «интуицию».

Очень важной задачей является автоматизация процессов исследований. Проблема подобных современных автоматизированных систем существенна как для гуманитарных, так и для физико-математических наук. В математике, например, делаются попытки создать автоматизированную систему доказательства теоремы.

Известен факт, когда математик одну теорему доказывал на 280 страницах. Эту работу немногие могли дочитать до конца, но и те, кто дочитал, не могли с уверенностью сказать, что в ней нет ошибок и что теорема действительно доказана.

Опасение, что не все, чем мы пользуемся при обосновании, скажем, теорем, доказано, является сегодня в математике вполне реальным. Можно хорошо ощущать математическую идею, ее можно понять, ею можно восхищаться, однако техника доказательства может подвести, особенно когда доказательство широкое. Систему доказательств — одну из отраслей математического творчества — можно с успехом контролировать с помощью ЭВМ.

Работа с ЭВМ полезна прежде всего самому математику: сумел объяснить машине принцип доказательства теоремы, значит сам хорошо разобрался в ней, не сумел — не доказал ее.

Для обоснования математических выкладок нужен специальный язык, ориентированный на математику. Необходима формализация понятия очевидности, создание машинного алгоритма, который доказывал бы очевидные вещи. Ведь такие понятия, как «машинная очевидность» и «человеческая очевидность», абсолютно различны. Аспекты очевидности немного смещены, и это нужно учитывать при создании формализованного языка. Математик иногда пытается подробно «разжевать» какое-то положение, которое абсолютно ясно машине. В машине могут быть сконцентрированы знания, которые по крупицам «рассыпаны» в целом коллективе научных работников.

Для решения проблемы автоматизации доказательства теоремы мы разрабатываем (кстати, впервые в мире) человеко-машинную систему доказательства, которая даст возможность значительно ускорить творческий процесс. Причем наиболее творческие элементы процесса оставляются человеку, а поиск в заданном направлении, оформление решения, различные выкладки передаются машине, которая делает это в сотни тысяч раз быстрее.

При этом само понятие «творчество» наполняется новым содержанием. Так, в шахматах вершиной творчества считалась комбинация, а сегодня ЭВМ по шахматным программам, составленным нами, создает комбинации лучше гроссмейстера. Таким образом, комбинация перестала быть творческим элементом. А вот оценка позиции или составление стратегических планов в шахматах намного труднее поддаются автоматизации, однако в принципе автоматизация возможна и в этих случаях.

Когда мы говорим о создании человеко-машинной системы для игры в шахматы, которую при условии, что она будет сделана хорошо, никто не сможет победить: ни человек, ни машина, то имеем в виду включение в

такую систему самого обыкновенного шахматиста, а не чемпиона. С помощью этой системы он победит любого гроссмейстера.

Перейдем к проблемам автоматизации процессов, связанных с творчеством, носящим прикладной характер.

При монтаже фильмов, особенно при создании мультипликационных лент, человеко-машинная система работает почти так же, как и система автоматизации проектирования.

Память машины сохраняет колоссальное количество различных элементов из предыдущих фильмов, нарисованных художниками-мультипликаторами и заложенных в нее. Кроме того, есть специальный «язык», с помощью которого можно «разговаривать» с машиной. Объясним это конкретнее. Художник монтирует кадры будущего мультфильма. Скажем, ему нужно нарисовать двухэтажный дом. Он нажимает соответствующие клавиши и, обращаясь к машине на понятном ей языке, вызывает на экране изображение дома. Это первый вариант. Художника этот вариант не устраивает, и он вновь нажимает клавишу, листает страницы специального альбома разных двухэтажных домов, находящихся в памяти машины. И в тот же миг изображения этих домов появляются на экране. Наконец художник останавливается на каком-то варианте. Но его не устраивают размеры дома, так как он несколько велик для кадра. Тогда дается команда: изменить масштаб. Изображение изменяется в соответствии с требованием и размещается именно там, где нужно, так как существует возможность передвигать его на экране. Затем может понадобиться, например, дерево в правой части экрана, а в левой — фигура человека. Изображение человека можно смонтировать по своему усмотрению. При этом прежде рисуется начальная позиция фигуры, затем конечная, а все промежуточные позиции нарисует машина.

Словом, художник освобождается от необходимости рисовать множество кадров, и изготовление мультипликационных фильмов с помощью такой человеко-машинной системы сокращается с полугода до нескольких недель, даже до трех-четырех дней, в зависимости от объема информационной базы машины. Если же художник не нашел в памяти машины нужное изображение, он берет электронное перо и тут же рисует новое изображение. Оно также остается в памяти машины и может быть использовано при создании другого фильма.

Немалую помощь может оказать ЭВМ и художникам в области прикладного искусства при создании декоративных комбинаций или разнообразных орнаментов в процессе разрисовки тканей. Здесь комбинаторные способности машины так велики, что человеку состязаться с ней нелегко. «Живой мольберт» соответственно запрограммированной машины может дать большое количество вариантов цветной гаммы, сочетающихся линий, штрихов различной густоты. Во всем этом процессе, однако, ведущая роль все же будет принадлежать человеку. Он может корригировать результат, предложенный машиной, в соответствии со своим вкусом, индивидуальным восприятием действительности.

Сложнее применять ЭВМ, когда мы выходим за рамки художественного творчества, носящего прикладной характер. Такими являются опыты, относящиеся к музыке. Человеко-машинные системы способны уже сегодня создать мелодии, грамотные с точки зрения композиции. Однако и тут ведущая роль принадлежит человеку. Машина по вашей программе «творит» музыку. После прослушивания, скажем, двухсот вариантов человек находит тот, который его удовлетворяет. В нашей прессе неоднократно освещались эксперименты, во время которых в различных аудито-

риях демонстрировались относительно простые мелодии, сочиненные ЭВМ, и мелодии, сочиненные профессиональными композиторами. При этом аудитории не сообщалось, какие именно мелодии «машинные». При закрытом анкетном опросе выяснилось, что слушатели не только не могли отличить первые от вторых, но чаще всего, а иногда и целиком отдавали предпочтение «машинным» мелодиям.

Разумеется, не может быть и речи о «замене» композитора машиной. Пусть музыковеды и композиторы решат, какие реальные возможности предоставляет им этот феномен. Ведь если посмотреть на опыты сочинения музыки машиной без предубеждения, то это действительно феномен, рожденный НТР! Этим открытием, разумеется, могут воспользоваться ремесленники, которые, впрочем, и без машин, сами фабрикуют свои однообразные, автоматизированно-однотонные мелодии. Но разве ссылкой на ремесленников можно закрыть данную проблему?

Большие перспективы открываются в развитии электронной музыки, демонстрирующей новые тембровые и оркестровые звучания. Такая музыка была написана для многих фильмов, например для «Соляриса». При ее сочинении композитор вводит в машину определенную мелодию, которую машина аранжирует, выдавая принципиально новые звучания, до этого не существовавшие. С помощью машины можно как угодно изменять тембр, получая такую окраску, которую не способен дать ни один инструмент.

Композиторов, работающих в области создания электронной музыки, привлекает именно неограниченность возможностей машины в этом направлении, а также то, что композитор становится одновременно и оркестрантом, он сам создает своеобразный оркестр без участия исполнителей. К тому же во много раз сокращается время написания музыкальных произведений. И, пожалуй, самое главное — появляются возможности написания музыки, которую каким-либо иным способом создать просто невозможно.

Но особенно поразительным является факт создания с помощью электронно-вычислительной техники сложных музыкальных произведений в стиле известных композиторов.

Уже существуют программы для ЭВМ в основном двух типов: для создания мелодий с учетом законов музыкальной композиции вообще и для имитации уже известных мелодий с учетом особенностей стиля автора. Вначале машина выявляет особенности музыкального стиля композитора, например Баха, а затем, вводя элемент случайности, не нарушающей особенностей стиля, сочиняет какую-то органную музыку, фугу или токкату. При этом даже у знатоков (если программа для ЭВМ составлена квалифицированно) не возникает сомнений, что они слушали Баха. Как могут быть использованы такие опыты? Конечно, не для всякого рода мистификаций, а прежде всего в помощь познанию и исследованию характерных особенностей стиля и «почерка» Баха или иного композитора. ЭВМ открывает и другие возможности в области теории и истории музыки. Еще на конгрессе Международной федерации по обработке информации (1968 г.) тема одного из докладов была связана с проблемой применения электронно-вычислительной техники в музыковедении. Ныне совместными усилиями программистов-математиков и музыковедов создаются машинные архивы музыки. Так, существует архив испанской музыки XVI столетия. Музыковед, изучающий историю музыки, может проверить свои обобщения, обратясь к колоссальной музыкальной памяти машины, в принципе неограниченной.

Немало споров вызывал вопрос о том, в какой степени ЭВМ может «создавать» литературные произведения. Конечно, сейчас можно составить такие программы, на основании которых машина «сочинит», например, стихотворение. Однако возникает вопрос: зачем, так как практически такого рода системы конкурировать с подлинными писателями не смогут. Да и трудности, которые нужно преодолеть при составлении соответствующих программ, себя не оправдывают. Сегодня всем очевидно, что интуиция человека, его жизненный опыт — элементы, недоступные для ЭВМ. У машин есть свои преимущества: они владеют колоссальной скоростью выполнения поручаемых им операций. Современная ЭВМ выполняет миллион арифметических операций за секунду, но не владеет гибкостью мышления, вытекающей из каких-то интуитивных источников.

Но есть проблемы, над которыми стоит работать. Это прежде всего автоматизация переводов с одного языка на другой. Как выглядел бы процесс перевода? Представим себе электронного переводчика с двумя специальными экранами. На один из них подается текст оригинала, на другой — машинный перевод. Кроме этого, устанавливается специальный читающий автомат, который соответствующим образом настраивается на текст, точнее — на шрифт. К сожалению, мы еще не имеем таких универсальных автоматов, которые могут читать любой шрифт с большой скоростью.

Представляем машине первую страницу-образец, написанную на том или ином языке. Машина, читая ее, делает необходимые вычисления и настраивает свою читающую часть (устройство) именно на данный шрифт, затем она с большой скоростью прочитывает весь текст и запоминает его. После этого работает примитивный алгоритм перевода. Понятно, по качеству такой перевод не может конкурировать (особенно если это художественный перевод) с переводом, выполненным специалистом. Но возможности улучшения полученного перевода существуют. Допустим, машине необходимо перевести русский текст на английский язык. Вначале, особенно когда машина еще не стала «профессиональным» переводчиком, может получиться так, что она не справится с каким-то словом, еще не зная его английского значения. На экране идет английский текст, а в нем — одно непереведенное русское слово. Писатель-переводчик следит за этим. У него есть печатная машина, соединенная с соответствующим экраном, и световой карандаш. Увидев непереведенное русское слово, писатель переводит его и впечатывает в текст. Таким образом, переводчик заполнил пробел, сделанный машиной. Когда перевод готов, его нужно отредактировать. Что это означает? Какие редакционные операции делают современные дисплеи? У переводчика, как упоминалось, есть специальный световой карандаш, пользуясь которым можно заменять или вычеркивать те или иные слова. Вместо одного или нескольких слов можно вмонтировать другие.

Весьма продуктивно можно использовать ЭВМ с целью рационализации труда литературоведа и искусствоведа. Известны случаи, когда при посредстве ЭВМ решался вопрос об авторстве анализируемого произведения, вопрос, бывший спорным на протяжении длительного времени. Вследствие колоссальной скорости чтения и практически безошибочной памяти ЭВМ сразу же запоминает особенности того или иного писательского стиля и сравнивает его со стилем исследуемого текста. Таким способом с достаточной убедительностью была подтверждена мысль о том, что «Илиада» и «Одиссея» написаны одним автором — Гомером.

ЭВМ могут использоваться в литературоведении и искусствознании с различными справочно-библиографическими и информационными целя

ми: для составления библиографий, автоматического индексирования, накопления и семантического поиска информации. Широкие возможности открываются в составлении словарей литературного языка, словарей рифм, синонимов и т. д.— не только с целью исследования творчества, но и в помощь писателю.

Особая область использования электронных опознающих систем — текстология, чтение черновых рукописей классиков, а также экспертное установление авторства того или иного произведения путем сравнения почерков. При этом необходимо составление «азбук» — образцов почерков различных писателей. Таким образом, развитие кибернетической текстологии не только увеличит достоверность анализа «не опознанных» в отношении авторства произведений, но и освободит ученого от одного из самых сложных и утомительных этапов его работы, важного не только для изучения процесса создания литературного произведения, но и при подготовке к печати академических собраний сочинений, учитывающих кроме основных и черновые варианты.

Нет сомнения, что по мере развития и усовершенствования электронной техники увеличатся возможности ее применения в самых разных областях, в том числе и в тех, которые связаны с литературным трудом. В принципе возможности работы системы на основе «общения человека с машиной» исключительно широки.

В истории кибернетики было немало сомнений в возможностях ЭВМ. Так, кое-кто уверял, что машина никогда не смоделирует условных рефлексов. Сегодня даже ученики, знакомые с кибернетикой, могут составить программу поведения животных в определенных условиях, а каких-то пять лет назад подобное казалось невозможным. Моделирование условных рефлексов — дошкольный уровень кибернетики.

Сейчас речь идет о том, что сегодня целесообразнее всего доверить машине.

Человеку свойственно естественное стремление противодействовать сведению его деятельности к механизированным операциям. Если, например, построить машину для разгадывания кроссвордов, то множество людей все равно будут разгадывать их своими силами. Стремление к интеллектуальному труду, к художественному творчеству заложено в самой природе человека. Поэтому в дальнейшем нужно, усовершенствуя машины, стремиться прежде всего к тому, чтобы способствовать с их помощью развитию творческих сил человека, а не угнетению их. Это возможно лишь тогда, когда автоматизацией творческих процессов будут заниматься люди, понимающие суть творчества. Случайный в искусстве человек не сможет даже с помощью ЭВМ создать что-либо ценное. Наивным было бы считать, что чуть ли не каждый с помощью ЭВМ сможет создавать стихи или музыкальные произведения.

Возникает вопрос, когда же автоматизированные системы (если они так удобны и нужны) можно будет широко применять с целью облегчения и ускорения литературного труда, освобождения времени и сил тех, кто им занимается, для непосредственно творческой деятельности?

Сегодня автоматизированные системы нужнее всего в области науки и конструирования. Без их внедрения немыслимы дальнейшие стремительные темпы технического прогресса. И, понятно, народные средства будут вложены прежде всего в создание таких систем. Задача ученых — непрерывно совершенствовать машины, чтобы можно было справиться наконец и с такими сложными проблемами, как создание «искусственного интеллекта». Прогресс в областях, рассмотренных в нашей статье, будет спо-

собствовать удешевлению пока еще дорогостоящей электронной техники: предполагается, что в недалеком будущем будут созданы ЭВМ, равные по стоимости цветному телевизору. Однако это не значит, что уже сегодня не следует работать над проблемами использования ЭВМ не только в технике, но и в сфере художественного творчества и гуманитарных наук. Ученые-гуманитарии в сотрудничестве со специалистами-кибернетиками могут расширять уже начатые работы, накапливать опыт, который затем позволит быстрее продвинуться вперед. Кое-что в этом направлении уже сделано сегодня.

Цель данной статьи — способствовать активизации интереса литературоведов, искусствоведов, практиков художественного творчества к проблеме «Кибернетика и творчество», способствовать их контактам с представителями точных наук. Именно на этом пути возможны реальные успехи в решении поставленных задач.

ТЕОРИЯ ОБУЧЕНИЯ ОДНОГО КЛАССА ДИСКРЕТНЫХ ПЕРСЕПТРОНОВ

(Журнал вычислительной математики и математической физики.— 1962.— № 2)

В работе [1] Розенблатт ввел одну интересную модель обучающегося автомата, названную им персептроном. Персептрон, по замыслу Розенблатта, должен был давать наглядное представление о механизме обучения распознаванию зрительных образов, являющемся одним из характерных примеров приспособительных свойств мозга. Теория обучения и самообучения в персептронах, развитая в работе [1], несмотря на использование математического аппарата, носит чисто качественный характер и не может служить основой ни для конкретных количественных оценок, ни для строгого доказательства особенностей асимптотического поведения персептрона.

В работе [2] Розенблатт несколько изменил определение персептрона и провел экспериментальное исследование его свойств. Аналогичные работы были выполнены также рядом других авторов (см., например, [3]). Введенное Розенблаттом изменение состояло в замене непрерывных моделей нейронов, применявшихся в работе [1], дискретными моделями.

Достаточно строгая теория обучения дискретных персептронов (т. е. персептронов, использующих дискретные модели нейронов) была развита Джозефом [4]. Однако эта теория не обладает нужной для реальных количественных расчетов степенью наглядности и, самое главное, применяет статистический подход лишь к анализу первоначальной конструкции персептрона, а не к динамике самого процесса обучения. Вследствие этого теория Джозефа становится фактически неприменимой в том случае, когда речь идет не о конкретных обучающих последовательностях, а о множестве таких последовательностей, обладающем теми или иными статистическими свойствами.

Предлагаемый в данной работе вариант теории обучения дискретных персептронов устраняет указанные недостатки и может служить основой для решения всех возникающих на практике задач, связанных с определением статистических закономерностей поведения дискретных персептронов достаточно широкого класса. Далее приводится ряд примеров применения предлагаемой теории для решения конкретных задач о поведении персептронов, принадлежащих к более общему классу, чем класс α-систем, рассмотренный Джозефом.

Отметим, что развиваемая здесь теория относится лишь к случаю дискретных персептронов. Тем не менее получаемые на ее основе качественные выводы в значительной мере можно отнести и к непрерывным персептронам, а также к другим известным в настоящее время моделям обучения распознаванию образов, например к адаптру Робертса [6] и к пандемониуму Селфриджа [6]. Этих выводов оказывается достаточно, в частности, чтобы установить принципиальное отличие всех указанных моделей от реального механизма обучения распознаванию образов, имеющегося в

мозгу человека. Факт неспособности персептрона моделировать многие важные особенности биологической организации обучения распознаванию образов был вынужден признать и сам Розенблатт [2]. В связи с этим в конце данной статьи высказываются некоторые предложения по изменению как структуры самого персептрона, так и методики его обучения. Реализация этих предложений позволит, как нам кажется, в значительно большей степени, чем это доступно персептрону, приблизиться к моделированию реальных процессов, происходящих в мозгу человека.

Прежде чем переходить к построению теории обучения в персептронах, напомним принадлежащие Розенблатту и Джозефу [1—3] определения понятий персептрона и класса персептронов.

Задавая персептрон, мы должны прежде всего задать так называемую сетчатку, иными словами, задать некоторое конечное множество чувствительных элементов, которые будем называть рецепторами. Обычно сетчатка задается как множество точек плоскости, причем чаще всего имеют дело с так называемой правильной сетчаткой, т. е. с множеством точек вида $(a + ic, b + jc)$, где $c \neq 0$, i пробегает множество значений $0, 1, \ldots, n-1$, и j — множество значений $0, 1, \ldots, m-1$. В этом случае принято говорить, что мы имеем правильную прямоугольную $(n \times m)$-сетчатку.

В ряде случаев оказывается удобным отождествлять противоположные края правильной сетчатки, полагая так называемую правильную тороидальную сетчатку. Мы будем представлять правильную тороидальную $(n \times m)$-сетчатку как множество точек плоскости с координатами $(a + ci, b + ci)$, где i и j — произвольные целые рациональные числа, рассматриваемые соответственно по модулям n и m.

Сетчатка предназначена для восприятия изображений из того или иного класса изображений. Изображение можно представлять себе в качестве некоторого проектируемого на сетчатку черно-белого рисунка, возбуждающего составляющие сетчатку рецепторы в соответствии с яркостью соответствующих им точек рисунка. На практике принято различать два вида рецепторов, которые мы будем называть соответственно непрерывными и дискретными (двоичными).

Непрерывный рецептор воспринимает всю возможную гамму яркостей, представляющую собой множество вещественных чисел от нуля (яркость фона рисунка) до некоторого положительного числа d (максимально возможной яркости точек рисунка). Сигнал, выдаваемый непрерывным рецептором, принимается равным яркости соответствующей ему точки рисунка.

Дискретный (двоичный) рецептор может выдавать лишь два сигнала, в качестве которых будем выбирать вещественные числа 0 и 1. При этом сами изображения можно рассматривать как имеющие всю непрерывную гамму яркостей. Все яркости, не превышающие некоторый устанавливаемый заранее порог, будут вызывать нулевой сигнал рецептора, а яркости, большие, чем порог,— единичный сигнал. Более удобно, однако, рассматривать в этом случае лишь двутональные изображения, состоящие либо из точек нулевой яркости (фон), либо из точек единичной яркости (собственно рисунок). Мы примем в дальнейшем именно последнюю точку зрения. Тогда, как нетрудно видеть, общее число различных изображений для любой сетчатки с дискретными рецепторами будет конечным. Для правильной прямоугольной (или тороидальной) $(n \times m)$-сетчатки с дискретными рецепторами число всех попарно различных изображений равно, очевидно, 2^{mn}. В дальнейшем мы все время будем ограничиваться лишь случаем двутональных изображений.

Изображения, с которыми оперирует персептрон, делятся на конечное число классов, называемых образами. В один и тот же образ зачисляются обычно изображения, которые объединяются человеком в одно общее понятие, например все квадраты, все горизонтальные линии или все изображения буквы «а».

В задачу персептрона входит определить принадлежность изображений к соответствующим классам. Для решения этой задачи персептрон имеет, кроме рецепторов, еще два вида элементов, называемых A и R-элементами.

A-элементы представляют собой упрощенные модели нейронов, из которых состоит человеческий мозг. В связи с этим мы будем их далее называть нейронами. В соответствии с характером используемых в персептроне рецепторов различают непрерывные и дискретные нейроны. Как те, так и другие нейроны имеют два вида входов, называемых возбуждающими и запрещающими. Каждый нейрон имеет конечное число входов и один выход; кроме того, с ним сопоставляется некоторое вещественное число, называемое весом данного нейрона. В качестве области значений за вес нейронов принимается множество всех вещественных чисел, независимо от того, идет речь о непрерывных нейронах или о дискретных.

Кроме веса, а также числа возбуждающих и числа запрещающих входов, нейрон характеризуется еще законом своего функционирования, определяющим выходной сигнал нейрона как функцию его входных сигналов и веса. Следует иметь в виду, что входы всех нейронов в персептроне подсоединяются к рецепторам, так что вырабатываемые рецепторами сигналы служат входными сигналами для нейронов.

Непрерывные нейроны, рассматривавшиеся первоначально Розенблаттом [1], имели закон функционирования вида $z = v(\Sigma x - \Sigma y)$, где z — выходной сигнал, v — вес нейрона, Σx — сумма сигналов, поступающих в нейрон по возбуждающим входам, а Σy — сумма сигналов поступающих по запрещающим входам (x, y, v и z — произвольные числа).

Закон функционирования дискретных нейронов задается обычно указанием некоторого целого рационального числа p, называемого порогом срабатывания нейрона или просто порогом. Если алгебраическая сумма $\Sigma x - \Sigma y$ возбуждающих и запрещающих входных сигналов меньше порога, то нейрон считается невозбужденным и выдает выходной сигнал, равный нулю. При достижении суммой $\Sigma x - \Sigma y$ порога и при превышении его нейрон возбуждается и выдает выходной сигнал, равный своему весу v (независимо от величины превышения суммы входных сигналов над порогом).

Дискретные нейроны с описанным законом функционирования удобно характеризовать тройкой целых чисел (k, l, p), первое из которых равно числу возбуждающих входов, второе — числу запрещающих входов, а третье — величине порога. Вес нейрона в дальнейших рассмотрениях будет всегда величиной переменной и потому в характеристику нейрона вводиться не будет. Дискретные нейроны указанного вида, имеющие одну и ту же характеристическую тройку чисел (k, l, p), условимся относить к одному и тому же типу независимо от возможного различия их весов.

Далее предполагается, что все нейроны любого данного персептрона принадлежат к одному типу. В персептроне, рассчитанном на различение k различных образов, множество всех нейронов разбивается на k попарно не пересекающихся групп (подмножеств), поставленных во взаимно-однозначное соответствие различаемым образам. Нейроны, принадлежащие группе, поставленной в соответствие i-му образу, будем для краткости называть просто нейронами i-го образа ($i = 1, \ldots, k$).

Входы каждого входящего в персептрон нейрона подсоединяются к рецепторам сетчатки. Будем предполагать при этом, что различные входы одного и того же нейрона подсоединяются к различным рецепторам. Выходы же нейронов подсоединяются к специальным сумматорам, называемым R-элементами, причем выходы нейронов одного и того же образа подсоединяются к одному и тому же сумматору, называемому сумматором этого образа.

Выходной сигнал сумматора любого данного образа равен сумме весов всех возбужденных нейронов этого образа. Если ни один из нейронов рассматриваемого образа не возбужден, то выходной сигнал соответствующего сумматора принимается равным нулю. Окончательным выходным сигналом всего персептрона считается тот образ, сумматор которого имеет наибольший выходной сигнал. В случае, когда максимальное значение выходного сигнала достигается одновременно сумматорами нескольких образов, выходной сигнал персептрона считается неопределенным.

Принимая в качестве входного сигнала всего персептрона проектируемое на его сетчатку изображение, в качестве реакции персептрона на этот сигнал получаем образ, к которому персептрон относит данное изображение. Разумеется, нельзя утверждать, что рассматриваемый персептрон осуществит правильную классификацию изображений в соответствии с заданным заранее разбиением множества изображений на различные образы. Это первоначальное разбиение задается человеком. Мы будем называть его исходной (или априорной) классификацией изображений, в отличие от фактической классификации, осуществляемой выбранным персептроном.

Необходимо еще задать некоторый процесс изменения характеристик персептрона, позволяющий по мере показа персептрону различных изображений приближать фактически производимую им классификацию к исходной. Этот процесс задается с помощью указания так называемого закона поощрения.

Для дискретных персептронов в качестве основного закона поощрения выберем несколько обобщенный закон поощрения в так называемых a-системах, рассматривавшихся Джозефом [3]. Этот закон, который будем называть (обобщенным) α-законом, вполне характеризуется заданием двух неотрицательных констант a и b, не равных нулю одновременно. Смысл данного закона поощрения состоит в том, что после каждого показа персептрону очередного изображения веса некоторых нейронов увеличиваются на величину равную a, а веса других уменьшаются на величину равную b (закон поощрения в α-системах Джозефа получается из (обобщенного) α-закона в случае, когда $a = 1$, $b = 0$).

Различают два режима функционирования персептрона с обобщенным законом поощрения. Первый режим, называемый режимом обучения, состоит в поощрении (увеличении веса на величину a) всех возбужденных нейронов того образа, которому принадлежит рассматриваемое на данном шаге изображение, и в штрафовании (уменьшении веса на величину b) всех возбужденных нейронов остальных образов. Ясно, что указание правильного образа, которому принадлежит данное изображение, должно осуществляться человеком-учителем, так как только ему известна исходная априорная классификация изображений.

Второй режим, называемый режимом самообучения, отличается от режима обучения только тем, что определение образа, которому принадлежит рассматриваемое изображение, производится самим персептроном: в качестве такого образа выбирается образ, который фактически был вы-

дан персептроном в ответ на показ данного изображения. Разумеется, при этом нет никакой гарантии, что выданный персептроном ответ будет правильным (в смысле исходной классификации изображений). Однако при соблюдении некоторых условий в случае неограниченного увеличения числа шагов в процессе самообучения персептрон может восстановить исходную классификацию изображений.

Кроме (обобщенного) α-закона поощрения, иногда оказывается целесообразным рассматривать еще два закона, которые будем называть соответственно (обобщенным) β-законом и (обобщенным) γ-законом. В обоих этих законах сохраняется принцип поощрения и штрафования, принятый в (обобщенном) α-законе. В дополнение к этому в β-законе на каждом шаге (как в режиме обучения, так и в режиме самообучения) происходит уменьшение веса всех нейронов (как возбужденных, так и невозбужденных) на величины, прямо пропорциональные их весам, с общим для всех нейронов коэффициентом пропорциональности β. В γ-законе осуществляется дополнительное (к операциям α-закона) изменение весов всех нейронов (как возбужденных, так и невозбужденных) на одну и ту же величину, выбираемую на каждом шаге так, чтобы сумма весов всех нейронов оказывалась всегда равной нулю.

В случае непрерывных нейронов (обобщенный) α-закон поощрения состоит в том, что любой нейрон правильного (априорно или с точки зрения персептрона) образа увеличивает свой вес на величину произведения q константы a на суммарный входной сигнал нейрона: $q = a\,(\Sigma x - \Sigma y)$. Аналогично веса нейронов всех остальных образов уменьшаются на величину $b\,(\Sigma x - \Sigma y)$ (свою для каждого отдельного нейрона). Дополнения, отличающие β- и γ-законы друг от друга, остаются такими же, как в дискретном случае.

При построении теории обучения и самообучения персептронов часто оказывается целесообразным рассматривать не отдельные из них, а некоторые классы персептронов. Классом персептронов будем называть множество персептронов, которые могут отличаться друг от друга лишь способом соединения нейронов с рецепторами и начальными весами нейронов. Все остальные характеристики персептронов, входящих в один и тот же класс, предполагаются одинаковыми. К числу этих характеристик относятся вид рецепторов и нейронов, общее число рецепторов и структура сетчатки, множество изображений и множество образов, исходная классификация изображений (распределение их по образам), число нейронов каждого образа и, наконец, закон поощрения.

Что же касается способа соединения нейронов с рецепторами и начальных весов нейронов, то эти характеристики считаются случайными и характеризуются (в пределах выбранного класса) некоторыми законами распределения. Иными словами, класс персептронов рассматривается не как абстрактное их множество, а как множество с заданным на нем полем вероятностей, определяющим вероятность выбора того или иного конкретного представителя рассматриваемого класса. Можно, таким образом, считать, что задание класса определяет некоторый случайный персептрон.

Начальные веса нейронов обычно считаются при этом независимыми случайными величинами, имеющими один и тот же закон распределения. Точно так же способ соединения каждого нейрона с сетчаткой предполагается не зависящим от соединения остальных нейронов, а с каждым возможным способом соединения отдельного нейрона с сетчаткой сопоставляется вероятность этого способа, общая для всех нейронов. При этом под-

соединение всех нейронов персептрона (из рассматриваемого класса персептронов) к сетчатке трактуется как серия независимых опытов, характеризуемых указанными вероятностями.

Сочетая вероятностную характеристику для способа соединения нейронов с сетчаткой с законом распределения начальных весов нейронов, приходим к искомому закону распределения в классе персептронов. Один из наиболее часто встречающихся законов распределения получается, когда все начальные веса детерминированы и равны одному и тому же числу (чаще всего нулю), а подсоединение всех входов любого данного нейрона производится независимо друг от друга на основании того или иного закона распределения (чаще всего равномерного), заданного непосредственно на сетчатке.

При построении теории обучения персептронов приходится рассматривать так называемые обучающие последовательности и классы обучающих последовательностей. Обучающая последовательность — это конечная последовательность изображений, показанных персептрону одно за другим в процессе его обучения или самообучения. Общее число показанных изображений (включая повторения) называется длиной обучающей последовательности. Классом обучающих последовательностей называется множество всех последовательностей одной и той же длины, в котором задан закон распределения, определяющий вероятность выбора любой данной последовательности рассматриваемого класса.

Чаще всего такой закон распределения получается с помощью приписывания любого изображения из рассматриваемого множества изображений на каждом шаге обучения, причем обычно рассматривается случай, когда эти вероятности одинаковы на всех шагах, т. е. когда обучающая последовательность представляет собой серию независимых экспериментов по выбору изображений с приписанными каждому из них неизменными вероятностями. Далее будем ограничиваться именно этим случаем.

Эффективность обучения в данном классе A персептронов с помощью данного класса B обучающих последовательностей определяется как вероятность правильного опознавания очередного изображения p, подаваемого на случайно выбираемый из класса A персептрон после предварительной подачи на него обучающей последовательности, случайно выбранной из класса B. При этом различают два вида эффективности. Так называемая полная эффективность обучения получается тогда, когда выбор изображения p производится случайно (с фиксированными заранее вероятностями появления различных изображений, использованными при установлении закона распределения в классе обучающих последовательностей). Эффективность обучения по одиночному изображению q получается, когда в качестве очередного изображения персептрону предлагается распознать именно изображение q.

На практике достаточно часто приходится рассматривать случай, когда все изображения в известном смысле равноправны как по отношению к рассматриваемому классу персептронов, так и в отношении равенства вероятностей появления этих изображений в обучающих последовательностях. В этом случае эффективность обучения по любому произвольно выбранному одиночному изображению будет столь же полно характеризовать способность персептронов к обучению, как и полная эффективность обучения.

Переходя к построению теории обучения персептронов, ограничимся рассмотрением лишь дискретных персептронов с (обобщенным) α-законом поощрения, работающих в режиме обучения (а не самообучения), не ого-

варивая это каждый раз. Теория обучения в этом случае получается наиболее простой и прозрачной, поскольку здесь оказывается возможным не следить за функционированием каждого отдельного нейрона, а ограничиться рассмотрением лишь некоторых интегральных характеристик.

В предлагаемом нами варианте теории такой интегральной характеристикой служит так называемый характеристический тензор персептрона. Подчеркнем, что применение термина «тензор» в этом случае не связано ни с какими закономерностями преобразования его компонент при изменении системы координат, а служит лишь названием некоторой целочисленной таблицы с тремя входами. Для описания такой таблицы введем определенную нумерацию всех изображений, принадлежащих рассматриваемому персептрону, числами от 1 до m и нумерацию всех образов, на которые подразделяются эти изображения, числами от 1 до g. Тогда характеристический тензор персептрона будет представлять собой совокупность компонент T_{ij}^k, где индексы i и j пробегают значения от 1 до m, а индекс k — от 1 до q. Через T_{ij}^k обозначается число нейронов k-го образа, которые возбуждаются как i-м, так и j-м изображениями.

Характеристический тензор класса персептронов определяется, по сути, аналогично. Различие между этими определениями лишь в том, что компоненты тензора T_{ij}^k будут на этот раз не детерминированными, а случайными величинами, законы распределения которых очевидным образом определяются законом распределения, характеризующим способ подсоединения нейронов к сетчатке.

Из приведенных определений непосредственно следует справедливость соотношения

$$T_{ij}^k = T_{ji}^k \quad (i, j = 1, \ldots, m; k = 1, \ldots, q). \tag{1}$$

Ясно также, что любой «диагональный» элемент тензора, например T_{ii}^k, представляет собой число нейронов того или иного (в данном случае k-го) образа, возбуждающихся под действием одного лишь i-го изображения. Отсюда непосредственно вытекает справедливость следующего неравенства:

$$T_{ii}^k \geqslant T_{ij}^k \quad (i, j = 1, \ldots, m; k = 1, \ldots, q). \tag{2}$$

Персептрон и класс персептронов будем называть симметричными, если компоненты характеристического тензора не зависят от верхнего индекса, т. е. если справедливо соотношение

$$T_{ij}^{k_1} = T_{ij}^{k_2} \quad (k_1, k_2 = 1, \ldots, q; i, j = 1, \ldots, m). \tag{3}$$

Верхний индекс в этом случае становится излишним, так что симметричные персептроны и их классы естественно характеризовать не трехвходовой таблицей (T_{ij}^k), а двухвходовой (T_{ij}), где $T_{ij} = T_{ij}^1 = T_{ij}^2 = \ldots = T_{ij}^q$.

Соответствующую двухвходовую таблицу условимся называть характеристической матрицей персептрона (или класса персептронов).

Введем еще одно обозначение. Для произвольной (конечной) последовательности изображений l через $U_i^k(l)$ будем обозначать выходной сигнал сумматора k-го образа, индуцируемый в рассматриваемом персептроне i-м изображением, показываемым после обучения персептрона обучающей последовательностью l. Через U_i^k обозначим соответствующий сигнал до начала процесса обучения, т. е. иными словами, сигнал $U_i^k(l)$ для случая, когда обучающая последовательность пуста (имеет длину, равную нулю).

Величины U_i^k будут, очевидно, детерминированными в случае выбора определенного персептрона и случайными в случае рассмотрения класса персептронов. Иногда оказывается целесообразным рассматривать также последовательность l как случайную, пробегающую рассматриваемый класс L обучающих последовательностей.

Особенностью α-закона обучения персептронов является своеобразное свойство коммутативности процесса обучения, выражаемое следующим предложением.

Теорема 1. *В персептроне (или в классе персептронов) с (обобщенным) α-законом поощрения выходной сигнал $U_i^k(l)$ сумматора k-го образа под действием i-го изображения после обучения любой последовательностью l не изменится, если в последовательности l будет произведена произвольная перестановка составляющих ее изображений. Это справедливо для любого изображения i и любого образа k.*

Действительно, входные сигналы нейронов k-го образа, индуцируемые i-м изображением, очевидно, не изменяются в процессе обучения, так что они остаются, в частности, одинаковыми после показа обучающей последовательности l и любой другой последовательности l'. Таким образом, изменение выходного сигнала сумматора в процессе обучения обусловлено исключительно изменением весов нейронов. Вследствие определения (обобщенного) α-закона изменение весов нейронов при показе любого изображения в процессе обучения (но, вообще говоря, не в процессе самообучения!) не зависит от того места, которое данное изображение занимает в обучающей последовательности. Поскольку же суммарное приращение веса любого нейрона в процессе обучения равно сумме приращений на каждом шаге этого процесса, то справедливость теоремы 1 тем самым полностью доказана.

Используя теорему 1, получаем возможность характеризовать любую обучающую последовательность l целочисленным вектором $\nu = (\nu_1, \nu_2, \ldots, \nu_m)$, i-я компонента которого (для любого $i = 1, 2, \ldots, m$) равна числу вхождений i-го изображения в последовательность l. Условимся называть этот вектор характеристическим вектором последовательности l. Класс обучающих последовательностей также можно задавать с помощью характеристического вектора. Однако компоненты вектора будут в этом случае, вообще, уже не детерминированными, а случайными величинами.

Для описания процесса обучения персептронов с (обобщенным) α-законом поощрения исходный персептрон (или класс персептронов) достаточно задавать лишь его характеристическим тензором (T_{ij}^k) и матрицей начальных сигналов сумматоров образов (U_i^k) $(i, j = 1, \ldots, m; k = 1, \ldots, n)$. Обучаемая же последовательность l (или класс обучающих последовательностей) задается своим характеристическим вектором $(\nu_1, \nu_2, \ldots, \nu_m)$. В общем случае все величины T_{ij}^k, U_i^k, ν_i будут случайными. На практике же чаще всего рассматривают различные частные случаи, когда те или иные из указанных величин детерминированы.

Чтобы не путать изображения и образы, условимся образы обозначать большими латинскими буквами, а изображения по-прежнему будем отождествлять с их номерами. Рассмотрим вопрос об определении выходных сигналов сумматоров образов $U_i^P(l)$. Легко понять, что при использовании (обобщенного) α-закона поощрения величина $U_i^{(P)}(l)$ будет представляться в виде суммы начального сигнала U_i^P и приращений весов всех нейронов P-го образа, возбуждающихся i-м изображением, на всех шагах процесса обучения.

Характеризуя класс обучающих последовательностей характеристическим вектором $(\nu_1, \nu_2, \ldots, \nu_m)$, нетрудно получить выражение для суммарного приращения величины $U_i^P(l)$, найденного за счет ν_j показов j-го изображения. Как следует из определения (обобщенного) α-закона с константами a и b, при каждом показе j-го изображения любой возбуждающийся этим изображением нейрон P-го образа увеличивает свой вес на величину b, если $j \in P$, и уменьшает его на величину b, если $j \notin P$. Общее же число нейронов P-го образа, участвующих в образовании выходного сигнала $U_i^P(l)$ и возбуждаемых j-м изображением, равно, очевидно, T_{ij}^P. Таким образом, суммарное приращение величины за счет ν_j показов j-го изображения выражается формулой $aT_{ij}^P\nu_j$, если $j \in P$, и формулой $bT_{ij}^P\nu_j$, если $j \notin P$. Непосредственно видна справедливость следующего предложения.

Теорема 2. *Пусть задан дискретный персептрон (или класс дискретных персептронов) с характеристическим тензором T_{ij}^P и матрицей начальных выходных сигналов сумматоров образов U_i^P $(i, j = 1, \ldots, m;\ P \in R)$. Если в рассматриваемом персептроне (классе персептронов) действует (обобщенный) α-закон поощрения с константами a, b, то после обучения последовательностью (или классом последовательностей) l с характеристическим вектором $(\nu_1, \nu_2, \ldots, \nu_m)$ для любого образа P и любого изображения i выходной сигнал $U_i^P(l)$ сумматора P-го образа под действием i-го изображения будет выражаться формулой*

$$U_i^P(l) = U_i^P + a\sum_{j\in P} T_{ij}^P\nu_j - b\sum_{j\notin P} T_{ij}^P\nu_j. \tag{4}$$

Для любого изображения i через P_i условимся обозначать образ, которому принадлежит изображение i в исходной классификации изображений. Используя это обозначение, нетрудно получить необходимое и достаточное условие для того, чтобы после обучения персептрон правильно классифицировал i-е изображение. Таким условием будет, очевидно, выполнение неравенств

$$U_i^{P_i}(l) > U_i^P(l) \quad \text{для всех } P \neq P_i. \tag{5}$$

С помощью соотношений (4) и (5) нетрудно рассчитать эффективность обучения персептрона в любом конкретном случае. Особенно простой вид эти соотношения принимают в случае симметричных персептронов. Действительно, поскольку в этом случае $T_{ij}^{P_i} = T_{ij}^P = T_{ij}$, то соотношения (4) и (5) можно записать в виде системы неравенств

$$U_i^{P_i} + a\sum_{j\in P_i} T_{ij}\nu_j - b\sum_{j\notin P_i} T_{ij}\nu_j > U_i^P + a\sum_{j\in P} T_{ij}\nu_j - b\sum_{j\notin P} T_{ij}\nu_j \quad (P \neq P_i). \tag{6}$$

В (6) слагаемые вида $\Sigma T_{ij}\nu_j$, для которых j не содержится ни в P_i, ни в P, будут входить как в левую, так и в правую части и поэтому взаимно уничтожатся. После их исключения приходим к более простым соотношениям, эквивалентным соотношениям (6):

$$U_i^{P_i} + (a+b)\sum_{j\in P_i} T_{ij}\nu_j > U_i^P + (a+b)\sum_{j\in P} T_{ij}\nu_j \quad \text{для всех } P \neq P_i. \tag{7}$$

Неравенства (7) являются необходимыми и достаточными условиями для правильной классификации i-го изображения симметричным персептроном с характеристической матрицей (T_{ij}) и начальными сигналами сумма-

торов образов $U_i^{P_i}$ после обучения последовательностью с характеристическим вектором $(\nu_1, \nu_2, \ldots, \nu_m)$. Эти неравенства можно еще больше упростить для персептронов с симметричными начальными условиями, т. е. для таких персептронов (или классов персептронов), для которых выполняются условия

$$U_i^P = U_i^Q \tag{8}$$

для всех $i = 1, 2, \ldots, m$ и для всех образов P и Q. Используя соотношения (8) и вспоминая, что вследствие определения (обобщенного) α-закона $a + b \neq 0$, приходим к следующему результату.

Теорема 3. *Пусть задан произвольный дискретный симметричный персептрон (или класс дискретных симметричных персептронов) с характеристической матрицей (T_{ij}) и с симметричными начальными условиями, в котором действует (обобщенный) α-закон поощрения. Тогда необходимые и достаточные условия правильности распознавания персептронов (классов персептронов) произвольного i-го изображения после обучения последовательностью (классом последовательностей) с характеристическим вектором $(\nu_1, \nu_2, \ldots, \nu_m)$ будут выражаться соотношениями*

$$\sum_{j \in P_i} T_{ij}\nu_j > \sum_{j \in P} T_{ij}\nu_j \quad \text{для всех } P \neq P_i. \tag{9}$$

Следствие. Эффективность обучения в симметричных дискретных персептронах с симметричными начальными условиями при выполнении в них (обобщенного) α-закона поощрения не зависит от выбора (неотрицательных) констант a и b, характеризующих этот закон.

Таким образом, при изучении симметричных дискретных персептронов с симметричными начальными условиями можно, не нарушая общности, вместо (обобщенного) α-закона поощрения с константами (a и b) пользоваться обычным α-законом поощрения с константами (1, 0).

При конкретных расчетах эффективности обучения в классах персептронов обычно принимается, что все нейроны подсоединяются к сетчатке независимо один от другого, причем вероятность α_{ij} такого подсоединения нейрона, что он будет возбуждаться как i-м, так и j-м изображениями, одна и та же для всех нейронов при любых фиксированных значениях i и j.

Если через T^P обозначить общее число нейронов P-го образа, то компоненту T_{ij}^P характеристического цензора рассматриваемого класса персептронов можно трактовать как число наступлений некоторого события, имеющего вероятность α_{ij} при T^P независимых испытаниях. Хорошо известно, что в таком случае математическое ожидание $E(T_{ij}^P)$ и дисперсия $D(T_{ij}^P)$ случайной величины T_{ij}^P выражаются следующими формулами:

$$E(T_{ij}^P) = T^P\alpha_{ij}, \quad D(T_{ij}^P) = \alpha_{ij}(1 - \alpha_{ij})T^P \quad (i, j = 1, \ldots, m; \ P \in R). \tag{10}$$

Величина T^P при достаточно больших значениях T может считаться нормально распределенной. Отметим еще, что в случае симметричных персептронов величины T^P для различных образов P равны друг другу. Поэтому будем обозначать их через T, опуская индекс P. Матрицу $\|\alpha_{ij}\|$ условимся называть основной вероятностной матрицей рассматриваемого класса персептронов.

Аналогично класс обучающих последовательностей K, образуемый с помощью случайного выбора изображения на каждом шаге обучения, независимо от изображений, выбранных на остальных шагах, можно характеризовать вероятностным вектором $(\beta_1, \beta_2, \ldots, \beta_m)$ рассматриваемого

класса. Для любого $i = 1, \ldots, m$ i-я компонента β_i этого вектора равна вероятности выбора в качестве изображения, показываемого на любом данном шаге обучения, i-го изображения. В таком случае i-я компонента ν_i характеристического вектора класса K представляет собой число наступлений события, имеющего вероятность β_i, при N независимых испытаниях, где N — длина обучающих последовательностей класса K (согласно определению класса обучающих последовательностей, все входящие в класс последовательности имеют одну и ту же длину).

Для достаточно больших значений N для любого изображения i величина ν_i может считаться нормально распределенной, а ее математическое ожидание и дисперсия задаются формулами

$$E(\nu_i) = N\beta_i, \quad D(\nu_i) = N\beta_i(1 - \beta_i) \quad (i = 1, \ldots, m). \tag{11}$$

Отметим, что случайные величины ν_i, как и случайные величины T_{ij}^D при различных i и j, не являются, вообще, независимыми, что создает дополнительные (хотя и чисто технические) трудности при вычислении вероятности правильного срабатывания персептрона по формулам (6) или (9). Впрочем, в ряде случаев с помощью введения некоторых добавочных предположений эти трудности удается обойти. Продемонстрируем сказанное на ряде примеров.

Пример 1. Рассматривается дискретный персептрон с нейронами типа (1, 1, 1), имеющий правильную квадратную $(n \times n)$-сетчатку и $2n$ изображений, в качестве которых выбирается n горизонтальных линий длины n, объединяемых в образ P, и n вертикальных линий длины n, объединяемых в образ Q. Все изображения имеют одинаковые вероятности (равные $^1/_2$) появления в обучающей последовательности. Будем предполагать персептрон A полным. Это означает, что как в множестве нейронов Q-го образа, так и в множестве нейронов P-го образа для любого способа подсоединения нейрона к сетчатке найдется в точности по одному нейрону, имеющему точно такое же соединение с сетчаткой. В персептроне A действует (обобщенный) α-закон поощрения с константами a и b, а начальные веса нейронов равны нулю.

Требуется найти эффективность обучения персептрона A в классе случайных обучающих последовательностей длины $2N$, содержащих точно N показов изображений первого образа и N показов изображений второго образа.

Решение. Персептрон A, очевидно, симметричен и поэтому будет полностью характеризоваться своей характеристической матрицей $\| T_{ij} \|$. Легко видеть, что нейрон тогда и только тогда возбуждается i-м изображением (вертикальной или горизонтальной линией), когда его возбуждающий вход подсоединен к рецептору, лежащему на соответствующей линии, а запрещающий вход — к рецептору, лежащему вне этой линии. Для любого данного i существует всего $n(n^2 - n) = n^2(n - 1)$ различных подсоединений такого рода. Ввиду полноты персептрона A получаем

$$T_{ii} = n^2(n - 1) \quad (i = 1, 2, \ldots, 2n). \tag{12}$$

Будем предполагать, что номерами от 1 до n обозначены горизонтальные линии (изображения образа P), а номерами от $n + 1$ до $2n$ — вертикальные (изображения образа Q).

По аналогии с тем, как было получено выражение для T_{ij}, находим еще два выражения:

$T_{ij} = 0$, если i и j — изображения одного и того же образа, (13)

$T_{ij} = (n - 1)^2$, если i и j — изображения различных образов. (14)

Обозначим через E единичную матрицу n-го порядка, а через D — квадратную матрицу порядка n, все элементы которой равны единице. Характеристическую матрицу M рассматриваемого персептрона представим в следующем виде:

$$M = \begin{vmatrix} n^2(n-1) & E(n-1)^2 D \\ (n-1)^2 D & n^2(n-1)E \end{vmatrix}. \tag{15}$$

Пусть $(\nu_1, \nu_2, \ldots, \nu_{2n})$ — характеристический вектор рассматриваемого класса обучающих последовательностей. Вследствие принятого условия компоненты этого вектора удовлетворяют условию

$$\nu_1 + \nu_2 + \cdots + \nu_n = \nu_{n+1} + \nu_{n+2} + \cdots + \nu_{2n} = N. \tag{16}$$

Вследствие теоремы 3 необходимое и достаточное условие правильного распознавания любого данного изображения i можно представить в виде

$$\sum_{j \in P_i} T_{ij}\nu_j > \sum_{j \in P} T_{ij}\nu_j, \tag{17}$$

или с учетом соотношений (12)—(14) и (16) — в виде

$$n^2(n-1)\nu_i > (n-1)^2 N. \tag{18}$$

Соотношение (18) представим в эквивалентном виде:

$$\nu_i > \left(\frac{1}{n} - \frac{1}{n^2}\right)N. \tag{19}$$

Вероятность появления i-го изображения в каждом из N показов представителей образа P равна $1/n$. Поэтому для математического ожидания и дисперсии величины ν_i получаем выражения

$$E(\nu_i) = \frac{1}{n}N, \quad D(\nu_i) = N\frac{1}{n}\left(1 - \frac{1}{n}\right). \tag{20}$$

Как хорошо известно из теории вероятностей, при достаточно большом N вероятность q_i выполнения неравенства (19) можно вычислить по формуле

$$q_i \approx 0{,}5 + \frac{1}{\sqrt{2\pi}}\int_0^k e^{-z^2/2}dz \quad (i = 1, \ldots, 2n), \tag{21}$$

где k — величина отношения уклонения правой части неравенства (19) от $E(\nu_i)$ к среднеквадратичному уклонению величины, равному квадратному корню из дисперсии. Иными словами,

$$k = \frac{1}{n^2}N : \sqrt{N\frac{1}{n}\left(1 - \frac{1}{n}\right)} = \sqrt{\frac{N}{n^2\left(1 - \frac{1}{n}\right)}} \approx \sqrt{\frac{N}{n^2}}. \tag{22}$$

Поскольку величина q_i не зависит от i, она будет совпадать с вероятностью q правильного распознавания персептроном A любого наугад выбираемого изображения после предварительного показа случайно выбираемой обучающей последовательности длины $2N$ из рассматриваемого класса последовательностей.

Величина же вероятности q есть не что иное, как полная эффективность обучения персептрона A в заданных условиях.

Приведем таблицу значений вероятности q для нескольких значений k:

N	n^2	$4n^2$	$9n^2$
k	1	2	3
q	0,841	0,977	0,999

Таким образом, чтобы уменьшить вероятность ошибки рассматриваемого персептрона до 0,1 % при случайном выборе обучающей последовательности, необходимо пользоваться последовательностями весьма большой длины, равной $18n^3$. В то же время из неравенства (17) видно, что эту вероятность можно свести к нулю (получив абсолютно точное распознавание) за счет показа каждого изображения точно по одному разу, т. е. с помощью последовательности, имеющей длину всего лишь $2n$. Этот пример демонстрирует невыгодность использования случайных обучающих последовательностей. Вместе с тем он свидетельствует о серьезных отличиях описанного механизма обучения от механизма обучения, реализующегося в мозгу человека.

Действительно, последний механизм обладает ярко выраженной способностью к экстраполяции опыта, т. е. к правильному распознаванию изображений, которые ни разу не появлялись в процессе обучения. В то же время описанный в рассмотренном примере персептрон не дает полной гарантии правильного распознавания изображений (при случайной организации процесса обучения) даже тогда, когда среднее число показа каждого изображения достигнет весьма значительной величины (порядка n^2).

Указанный вывод до некоторой степени связан, разумеется, со спецификой самого примера. Нетрудно заметить, однако, что при чисто случайном подсоединении (1, 1, 1)-нейронов к сетчатке (исключая подсоединения к одному и тому же рецептору обоих входов нейрона) математические ожидания компонент характеристической матрицы будут отличаться от компонент характеристической матрицы полного персептрона лишь постоянным множителем, несущественным с точки зрения вычисления эффективности обучения. Поэтому при случайном подсоединении нейронов к сетчатке наиболее вероятным поведением получающихся персептронов будет описанное выше поведение полного персептрона.

Таким образом, случайная организация связей нейронов с сетчаткой не может, вообще говоря, обеспечить хорошее качество функционирования персептронов. Из теоремы 3 следует, что способность персептрона к экстраполяции опыта увеличивается при увеличении тех компонент характеристической матрицы, индексы которых принадлежат одному и тому же образу, и при уменьшении тех компонент, индексы которых принадлежат различным образам.

Условимся говорить, что персептрон обладает абсолютной способностью к экстраполяции, если для любого образа P и любого изображения i из этого образа обучения любой последовательностью, содержащей изображение i не менее одного раза, он приводит к правильному распознаванию всех изображений данного образа.

Справедлив следующий результат.

Теорема 4. *Для того чтобы дискретный симметричный персептрон с симметричными начальными условиями, в котором действует (обобщенный) α-закон поощрения, обладал абсолютной способностью к экстраполяции, необходимо и достаточно, чтобы все компоненты T_{ij} характеристической матрицы персептрона, индексы которых принадлежат одному и тому же образу, были отличны от нуля, а все компоненты T_{ij}, индексы которых принадлежат различным образам, были равны нулю.*

Действительно, пусть условие теоремы выполнено. Тогда неравенство (9) будет выполняться, если хотя бы для одного изображения l из P_1 величина v_j отлична от нуля. Вследствие теоремы 3 это и означает, что рассматриваемый персептрон обладает абсолютной способностью к экстраполяции.

Предположим, что условие теоремы не выполнено. Это приводит к рассмотрению двух случаев: 1) для некоторого образа Q найдется принадлежащая ему пара изображений i, j такая, что $T_{ij} = 0$; 2) найдется пара изображений k, r, принадлежащих различным образам, такая, что $T_{kr} \neq 0$. В первом случае в силу теоремы 3 обучающая последовательность, составленная исключительно из изображений j, не приведет к правильному распознаванию изображения i. Во втором рассмотрим обучающую последовательность, составленную из одного изображения k и любого числа ν_r (большего, чем T_{kk}/T_{kr}) изображений r. Тогда применительно к распознаванию изображения k подстановка указанных значений в неравенство (9) приводит к неравенству $T_{kk} > \nu_r T_{kr}$. Ввиду выбора ν_r это неравенство неверно, что вследствие теоремы 3 означает невозможность правильного распознавания изображения k. Тем самым доказано, что в обоих случаях персептрон не будет обладать абсолютной способностью к экстраполяции, что и требовалось доказать.

Обычно при нумерации изображения, принадлежащие одному и тому же образу, нумеруются последовательными целыми числами. При этом условии характеристические матрицы симметричных персептронов естественным образом разбиваются на клетки, соответствующие различным образам. Абсолютная способность к экстраполяции достигается в том случае, когда эти матрицы клеточно-диагональные, а диагональные клетки не содержат нулевых элементов.

Такой вид характеристических матриц не всегда достижим в полной мере, однако даже сколько-нибудь удачное приближение к нему требует, как правило, отказа от вполне случайного соединения нейронов с сетчаткой. Получаемый в результате такого отказа эффект продемонстрируем на примере.

Пример 2. Найти эффективность обучения персептрона B, отличающегося от персептрона A из примера 1 лишь тем, что в нем оставлены только те нейроны, оба входа которых подсоединены к рецепторам, лежащим либо на одной горизонтальной, либо на одной вертикальной линии. Условия обучения остаются такими же, как в примере 1.

Решение. Персептрон B, как и персептрон A, будет, очевидно, симметричным. Нетрудно подсчитать, что элементы его характеристической матрицы задаются следующими соотношениями:

$$T_{ii} = n(n-1), \quad T_{ij} = 0 \quad (i, j = 1, \ldots, 2n, \ i \neq j).$$

Условие правильного распознавания i-го изображения будет иметь вид $T_{ii}\nu_i > 0$, или, что то же, $\nu_i > 0$. Иначе говоря, для правильного распознавания i-го изображения необходимо и достаточно, чтобы оно было хотя бы раз показано персептрону в процессе его обучения.

При N случайных показах изображений одного образа вероятность непоявления в обучающей последовательности i-го изображения равна, очевидно, $(1 - 1/n)^N \approx e^{-N/n}$, а полная эффективность обучения выражается числом $1 - e^{-N/n}$. Чтобы уменьшить вероятность неправильного срабатывания персептрона до 0,1 %, аналогично тому как это было сделано в примере 1, достаточно положить $N = 7n$ или, иными словами, оперировать обучающей последовательностью длины $14n$. Напомним, что в примере 1 такая же эффективность обучения достигалась лишь на обучающих последовательностях длиной $18n^3$.

Интересно, что столь резкое увеличение эффективности обучения достигается не за счет усложнения, а за счет упрощения персептрона, поскольку персептрон B получается из персептрона A с помощью выбрасы-

вания большого числа плохо присоединенных к сетчатке нейронов. Легко подсчитать, что общее число нейронов в персептроне A равно $2n^2(n^2 - 1)$, а в персептроне B — лишь $4n(n-1)$. Это еще раз свидетельствует о несовершенстве обучающего механизма персептрона и его существенном отличии от процессов обучения, протекающих в мозгу человека.

Рассмотрим еще один пример расчета эффективности обучения в классе персептронов.

Пример 3. Определить эффективность обучения класса C дискретных симметричных персептронов с симметричными начальными условиями, подчиняющихся (обобщенному) α-закону поощрения. Сетчатка, образы и изображения такие же, как и в примере 1. Число нейронов каждого из двух имеющихся образов равно N. Входы всех нейронов подсоединяются независимо друг от друга с равной вероятностью к любому рецептору сетчатки, исключая случай одновременного подсоединения обоих входов нейрона к одному и тому же рецептору. Обучающая последовательность содержит каждое из $2n$ изображений точно по одному разу.

Решение. Легко видеть, что компоненты T_{ij} характеристической матрицы класса C, у которой индексы i и j являются различными изображениями одного и того же образа, равны нулю. Условие правильного распознавания i-го изображения, задаваемое теоремой 3_α, в нашем случае можно записать в виде

$$T_{ii} > \sum_{j \in P_i} T_{ij}. \tag{23}$$

Легко видеть, что множества M_{ij} нейронов одного и того же образа, возбуждающихся как i-м, так и j-м изображением, при различных j, отличных от i, попарно не пересекаются. Все эти множества содержатся, разумеется, в множестве M_{ii}. Так как T_{ij} есть не что иное, как число элементов множества M_{ij}, то для выполнения неравенства (23) необходимо и достаточно, чтобы среди нейронов образа P_i нашелся хотя бы один нейрон, возбуждающийся i-м изображением, но не возбуждающийся никаким изображением противоположного (отличного от P_i) образа.

Из геометрии изображений непосредственно вытекает, что этому условию удовлетворяют нейроны, оба входа которых подсоединены либо к одной и той же вертикали (если i — горизонтальная линия), либо к одной и той же горизонтали (если i — вертикальная линия). Для любого фиксированного i из общего числа $n^2(n^2 - 1)$ различных подсоединений этому условию удовлетворяет лишь $n(n-1)$ подсоединений. Поэтому вероятность желательного подсоединения равна $n(n-1)/n^2(n^2-1) = 1/(n(n+1))$, а вероятность того, что такое подсоединение не будет иметь места ни для одного из N нейронов, равна $(1 - 1/n(n+1))^N \approx e^{-N/n(n+1)}$. Следовательно, полная эффективность обучения выражается формулой

$$r = 1 - e^{-N/n(n+1)}.$$

Если число нейронов каждого образа равно $7n(n+1)$, т. е. примерно в семь раз больше общего числа рецепторов, то вероятность неправильного срабатывания наугад выбирается из класса C персептрона и после обучения с помощью показа всех изображений по одному разу будет равна e^{-7}, что равно приблизительно 0,001.

Как уже отмечалось выше, построенная теория обучения персептронов свидетельствует о коренных отличиях реализуемого ими механизма процесса обучения от реального процесса обучения, происходящего в мозгу

человека. Переход от дискретных нейронов к непрерывным, как и замена α-закона поощрения β- или γ-законом, это положение существенно не меняет. Частично данный вопрос можно решить за счет добавления к реализуемым в персептроне процессам процесса пересоединения нейронов, мешающих процессу обучения или недостаточно способствующих ему.

Можно предусмотреть, например, периодическую проверку весов нейронов и случайные пересоединения нейронов с малым весом. Механизмы такого рода реализуются в схемах адапта Робертса и пандемониума Селфриджа [5, 6], позволяющих увеличивать коэффициент использования оборудования и уменьшать число нейронов, достигающее в схемах персептронов с чисто случайными соединениями нейронов непомерно больших значений.

Вместе с тем указанная мера совершенно недостаточна для объяснения такой особенности приспособительных функций мозга, как использование тех или иных признаков, выделенных на уже изученных образах, для ускорения процесса обучения распознаванию новых образов, содержащих все или часть этих признаков. Легко понять, что подобный процесс можно реализовать в многоступенчатых персептронах, т. е. в таких схемах, у которых сумматоры образов персептрона низшей ступени используются в качестве рецепторов для персептрона следующей ступени. При этом персептроны низших ступеней обучаются распознаванию отдельных свойств образов, а персептроны высших ступеней — распознаванию наборов этих свойств. Соответствующие изменения и усложнения законов поощрения можно выполнить многими различными способами, на описании которых мы останавливаться не будем.

Введение перечисленных усовершенствований не позволяет, однако, приблизиться к моделированию еще одной важной особенности, присущей мозгу, а именно к установлению инвариантности всех образов по отношению к их движению и изменению размеров на основе ограниченного опыта, использующего только небольшую часть всех образов. Для достижения успеха и в этом направлении необходимо изменить не только конструкцию персептрона, но и методику самого процесса обучения. С этой целью для распознающего устройства вводится возможность вмешательства в организацию обучающей последовательности.

Если, например, распознающему устройству A в качестве представителей того или иного образа показываются несколько различных изображений, то устройство A должно обладать возможностью повторять демонстрацию этих изображений столько раз, чтобы обеспечить в дальнейшем их правильное распознавание. Более того, устройству должна быть предоставлена возможность повторения показа тех же изображений, подвергнутых таким изменениям, которым обычно подвергается изображение предмета на сетчатке глаза при изменениях взаимного положения глаза и рассматриваемого предмета.

Можно, разумеется, не вводить описанную обратную связь, позволяющую распознающему устройству изменять обучающую последовательность. Вместо этого сами обучающие последовательности нужно строить так, чтобы после показа того или иного изображения увеличивалась бы вероятность показа на следующем шаге этого же изображения, рассматриваемого, быть может, лишь под другим ракурсом, либо, по крайней мере, изображений, принадлежащих этому же образу. Иными словами, при обучении распознающих устройств необходимо отказаться от построения процесса обучения по схеме независимых испытаний и перейти к более сложным схемам, описываемым марковскими цепями.

Предполагаемые изменения методов построения обучающихся последовательностей намного улучшают функционирование распознающих устройств в режиме простого обучения. Что же касается режима самообучения, то для него эти изменения имеют принципиальное значение, поскольку лишь на таком пути можно надеяться, что классификация изображений, производимая самообучающимися устройствами, будет соответствовать исходной классификации, производимой человеком. Ясно, что описание подобных процессов требует гораздо более сложного математического аппарата, чем тот, который был использован в данной работе.

СПИСОК ЛИТЕРАТУРЫ

1. *Rosenblatt F.* Two theorems of statistical separability in the perceptron, Symp. Mechaniz. Thought Proc. Teddington, England, 1958, Paper 1—3, 3—32.
2. *Rosenblatt F.* Perception simulation experiments // Proc. IRE.— 1960.— 48, N 3.— P. 301—309.
3. *Murray A. E.* A review on the perceptron programm // Proc. Nat. Electronics Conf., Chicago 111.— 1959, 15.— P. 346—356.
4. *Joseph R. G.* On predicting perceptron performance // IRE, Intern. Conv. Rec.— 1960.— N 2.— P. 71—72.
5. *Roberts L. G.* Pattern recognition with an adaptive network // IRE, Intern. Conv. Rec.— 1960.— N 2.— P. 66—70.
6. *Selfridge O. G.* Pandemonium: a paradigm for learning. Symp. Mechaniz. Thought Proc. Teddington, England, 1958, Paper 3—4, 3—16.

К ВОПРОСУ О САМООБУЧЕНИИ В ПЕРСЕПТРОНЕ

(Журнал вычислительной математики и математической физики.— 1962.— № 6)

В работе [1] исследовано поведение одного класса дискретных персептронов (так называемых α-персептронов) в режиме обучения. Характерной чертой режима обучения является наличие учителя, которому известна правильная классификация изображений. В данной статье исследуются некоторые вопросы, связанные с поведением дискретных α-персептронов в режиме самообучения. В этом случае учитель отсутствует, а процессы самоорганизации, приводящие к изменению производимой персептроном классификации изображений, определяются введенной в схему персептрона положительной обратной связью.

Хорошо известно, что анализ поведения персептронов в режиме самообучения, сделанный Розенблаттом [2], весьма далек от какого-то ни было строгого математического анализа. Отсутствие строго доказанных предложений в этой области привело к тому, что некоторые авторы (особенно в публикациях научно-популярного характера) приписывают самообучению персептронов многие свойства, которыми оно в действительности не обладает и не может обладать. Чтобы хотя бы частично решить эту проблему, в этой статье предпринята попытка более строго рассмотреть некоторые задачи, связанные с изучением самообучения в персептронах. На основании такого рассмотрения нетрудно сделать ряд выводов, очерчивающих границы действительных возможностей, заложенных в самообучении персептронов. При дальнейшем изложении будем предполагать известными основные определения и результаты работы [1].

Рассмотрим дискретный симметричный α-персептрон, рассчитанный на распознавание двух образов P и Q. Условимся в качестве единого выходного сигнала персептрона рассматривать разность выходных сигналов сумматоров P-го и Q-го образов:

$$V_i(l) = U_i^P(l) - U_i^Q(l). \tag{1}$$

Здесь, как и в [1], индекс i пробегает все изображения (как P-го, так и Q-го образов), а l — любая последовательность изображений, показанная персептрону в процессе его самообучения.

Как и в случае обучения, нетрудно показать, что функционирование симметричного персептрона в режиме самообучения определяется суммой $a + b$ констант поощрения и штрафа, а не этими константами, рассматриваемыми отдельно. Имея в виду также возможность произвольного изменения масштабов, не нарушая общности, можем предположить, что $a = 1$, а $b = 0$. Далее мы всегда будем исходить из этого допущения.

Как и в работе [1], через $\| T_{ij} \|$ обозначаем характеристическую матрицу персептрона и с учетом определения α-закона поощрения (см. [1]) легко приходим к формуле

$$V_i(lj) = V_i(l) + T_{ij} \operatorname{sign} V_j(l). \tag{2}$$

Формула (2) справедлива для любой пары изображений i, j и любой последовательности изображений l. Функция sign x, как обычно, для положительных значений x принимается равной плюс единице, а для отрицательных — минус единице. Ясно, что в случае нулевого значения величины $V_j(l)$, как следует из закона поощрения (положительной обратной связи), величина sign $V_j(l)$ в формуле (2) должна быть неопределенной. Для избежания этой неопределенности далее будем по определению считать нуль положительной величиной, так что sign $0 = +1$.

С учетом указанного видоизменения в определении функции sign x формулу (2) будем рассматривать как способ рекуррентного задания вектора $V(l) = (V_1(l), ..., V_m(l))$, определяющего выходные сигналы персептрона под действием любого изображения $j = 1, ..., m$ после подачи на вход персептрона последовательности изображений l. Начальное значение этого вектора $V(0) = (V_1(0), ..., V_m(0))$ предполагается при этом заданным. Изображение j причисляется персептроном к образу P или Q в соответствии с тем, положительна или отрицательна соответствующая компонента $V_j(l)$ рассматриваемого вектора (напомним, что нуль, по принятому соглашению, считается положительным числом).

Поскольку все значения T_{ij} являются целыми (и к тому же неотрицательными) числами, задача расчета персептрона в режиме самообучения по сути сводится к задаче случайного блуждания по дискретной решетке в пространстве с числом измерений, не превышающим m. Предполагая, что показ изображений в процессе самообучения производится по схеме независимых испытаний, легко видеть, что вероятности переходов из любой такой решетки определяются лишь набором знаков координат этой точки.

Как нетрудно понять, набор знаков координат любой точки решетки определяет также классификацию изображений, производимую персептроном, который имеет в качестве вектора $V(l)$ своих выходных сигналов радиус-вектор этой точки. С точки зрения теории персептрона интерес прежде всего представляет предельное распределение знаков координат вектора $V(l)$ при неограниченном увеличении длины обучающей последовательности l. Из приведенного выше рассмотрения следует, что требуемое распределение получается из предельного распределения для марковской цепи, соответствующей описанному выше блужданию по дискретной решетке.

Поскольку указанная цепь имеет бесконечное число состояний, нахождение предельного распределения в общем случае является достаточно сложным. Можно, однако, указать ряд случаев, когда задача нахождения предельного распределения легко сводится к исследованию марковской цепи с конечным числом состояний.

В качестве примера рассмотрим дискретный симметричный α-персептрон A, рассчитанный на распознавание $2n$ изображений, первые n из которых принадлежат образу P, а последние n — образу Q. Пусть, далее, для элементов характеристической матрицы персептрона A выполняются соотношения $T_{ij} = a > 0$, если i и j принадлежат одному и тому же образу, и $T_{ij} = 0$, если i и j принадлежат различным образам. Вследствие теоремы 4 из работы [1] рассматриваемый персептрон обладает абсолютной способностью к экстраполяции и, следовательно, наилучшим образом ведет себя в режиме обучения (обучается правильному распознаванию в результате показа хотя бы одного изображения из каждого образа). Предположим, что начальными условиями будут $V_i(0) = b > 0$ для $i = 1, 2, ..., m \leqslant n$ и $V_j(0) = -b$ для $j = m + 1, ..., n, ..., 2n$.

Из формулы (2) непосредственно следует, что $V_j(l) < 0$ для любой последовательности l при всех $j = n+1, n+2, \ldots, 2n$. Остальные же компоненты будут выражаться формулами $V_i(l) = b + ka$ для $i = 1, \ldots, m$ и $V_i(l) = -b + ka$ для $i = m+1, \ldots, n$, где k — разность между числом появления изображений, которым соответствуют положительные компоненты $V_i(l')$, и числом изображений, которым соответствуют отрицательные компоненты $V_i(l')$ (l' — соответствующая подпоследовательность последовательности l).

Предположим, что процесс самообучения совершается по схеме независимых испытаний с равными вероятностями появления всех изображений. Поскольку показ изображений одного образа в рассматриваемом случае никак не влияет на распознавание изображений второго образа, можно, не нарушая общности, предполагать, что в процессе самообучения участвуют только изображения образа P (изображениям образа Q всегда соответствует отрицательный выходной сигнал независимо от того, включаются они в процесс самообучения или нет).

Допустим, что b не делится на a, и через t обозначим целое число $[b/a] + 1$. Нетрудно понять, что для изучения функционирования персептрона A интерес представляют лишь те значения параметра k, которые заключены в замкнутом интервале $[-t, t]$. Действительно, если в процессе самообучения величина k хотя бы один раз достигает значения t, то в дальнейшем вследствие формулы (2) она может только возрастать, причем персептрон, начиная с этого момента, будет давать положительный выходной сигнал для всех изображений образа P (что соответствует правильной классификации). Аналогично, если параметр k примет значение $-t$, то в дальнейшем он сможет только убывать, а персептрон будет давать отрицательный выходной сигнал для всех изображений (что фактически означает отсутствие какой-либо классификации изображений, поскольку все изображения причисляются персептроном к одному и тому же образу).

Теперь, как легко видеть, предельное поведение персептрона A определяется марковской цепью с $2t+1$ состояниями $k = -t, -t+1, \ldots, -1, 0, 1, \ldots, t-1, t$. Вследствие принятого предположения о вероятностях появления изображений в процессе самообучения, для любого k, отличного от t или $-t$, вероятность перехода в состояние $k+1$ равна m/n, а вероятность перехода в состояние $k-1$ равна $(n-m)/n$. Из состояния t (как и из состояния $-t$) возможен переход только в это же состояние, поскольку с точки зрения функционирования персептрона все состояния при $k > t$ (соответственно при $k < -t$) не отличаются от состояния $k = t$ (соответственно от состояния $k = -t$).

Вводя обозначения $p = m/n$ и $q = (n-m)/n$, для рассматриваемой марковской цепи получаем матрицу переходных вероятностей

$$M = \begin{vmatrix} 1 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\ p & 0 & q & 0 & \ldots & 0 & 0 & 0 & 0 \\ 0 & p & 0 & q & \ldots & 0 & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & \ldots & p & 0 & q & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & p & 0 & q \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 1 \end{vmatrix}.$$

Эта матрица имеет единицу в качестве своего двукратного собственного значения. Вероятности перехода цепи в состояния t и $-t$ равны предельным переходным вероятностям p^{∞}_{t+1} и $p^{\infty}_{t+1,2t+1}$. Для вероятности $P^{\infty}_{t+1,i}$ по известной формуле Перрона находим выражение

$$p^{\infty}_{t+1,i} = \lim_{s\to\infty} \frac{d}{d\lambda} \frac{\lambda^s M_{i,t+1}(\lambda)}{\psi(\lambda)} \Bigg|_{\lambda=1}. \qquad (3)$$

Легко видеть, что $M_{i,t+1}(\lambda)$ для i, отличного от 1 и от $2t+1$, делится на $(\lambda-1)^2$ и поэтому $P^{\infty}_{t+1,i} = 0$ для этих значений i. Для $i = 1$ имеем $M_{1,t+1}(\lambda) = (\lambda-1)N_1(\lambda)$, а для $i = 2t+1$ $M_{2t+1,t+1}(\lambda) = (\lambda-1)\times$ $\times N_2(\lambda)$, где $N_1(\lambda) = p^t Q(\lambda)$, $N_2(\lambda) = q^t R(\lambda)$.

Из формулы (3) теперь легко получаем

$$p^{\infty}_{t+1,1} = cp^t, \quad p^{\infty}_{t+1,2t+1} = cq^t.$$

Поскольку все остальные предельные переходные вероятности в $(t+1)$-й строке равны нулю, из условия стохастичности матрицы предельных переходных вероятностей находим значение c: $c = 1/(p^t + q^t)$. Тем самым нами доказано следующее предложение.

При неограниченном продолжении процесса самообучения описанный выше персептрон A с вероятностью $p^t/(p^t + q^t)$ устанавливает правильную классификацию изображений и с вероятностью $q^t/(p^t + q^t)$ относит все изображения к одному и тому же образу.

Можно легко заметить, что рассмотренный пример, вообще, не может быть осуществлен в реальном персептроне, исключая тривиальные случаи: $m = n$, $p = 1$, $q = 0$ и $m = 0$, $p = 0$, $q = 1$. Это связано с тем, что при сделанных предположениях относительно характеристической матрицы все изображения одного и того же образа возбуждают одно и то же множество нейронов. Поэтому выходные сигналы, индуцируемые изображениями одного и того же образа, всегда, в том числе и в начальный момент, должны быть равны друг другу.

Нетрудно, однако, видеть, что, полагая $T_{ij} = a + \delta$ для всех $i = 1, \ldots$ $\ldots, 2n$ ($\delta > 0$) и не изменяя остальных элементов характеристической матрицы, мы получаем возможность удовлетворить введенным в примере начальным условиям. Вместе с тем, если δ существенно меньше, чем a, а t относительно невелико, то описанное в примере поведение персептрона будет служить хорошим приближением для его истинного поведения.

Рассмотрим теперь полный дискретный α-персептрон B с (1, 1, 1)-нейронами, с квадратной $(n \times n)$-сетчаткой, рассчитанный на распознавание двух образов P и Q. Образ P состоит из n горизонтальных линий, а образ Q — из n вертикальных линий; каждая из этих $2n$ линий составляет отдельное изображение. Как уже отмечалось в работе [1], персептрон B можно рассматривать как наиболее характерного представителя класса персептронов со случайными связями нейронов с сетчаткой. Согласно теореме 1 и следующему за ней следствию из работы Розенблатта [2], такие персептроны, будучи построенными на непрерывных нейронах, с вероятностью, сколь угодно близкой к единице, при самообучении должны стремиться к состоянию, в котором все изображения относятся к одному и тому же образу. Покажем, что для персептрона B подобное утверждение не выполняется.

Легко видеть, что для персептрона B начальные условия можно выбрать любыми. Нижней границей модулей начальных условий будем называть наименьшее из чисел $|V_i(0)|$ $(i = 1, 2, \ldots, 2n)$. При сделанных предположениях справедлив следующий результат.

Теорема 1. *Для любого сколь угодно малого положительного числа ε найдется такое число s, что в случае, когда нижняя граница модулей начальных условий превосходит s, персептрон B в режиме самообучения (с равновероятным появлением всех изображений) с вероятностью $p > 1 - \varepsilon$ сохраняет начальную классификацию изображений.*

Доказательство. Обозначим через N длину обучающей последовательности l, а через ν_i — число появлений i-го изображения ($i = 1, 2, \ldots, 2n$) в этой последовательности. Пусть $V_j(0) = x_j$ ($j = 1, 2, \ldots, 2n$); K_i — множество всех индексов j (изображений), относящихся к противоположному по сравнению с i образу и таких, что знак x_j совпадает со знаком x_i; L_i — множество всех индексов j, относящихся к противоположному по сравнению с i образу и таких, что знак x_j противоположен знаку x_i (нуль, как и ранее, считается при этом положительным числом).

Как было показано в [1], произвольный элемент T_{ij} характеристической матрицы персептрона B равен $n^2(n-1)$, 0 или $(n-1)^2$ в зависимости от того, совпадают ли индексы i и j, не совпадают, но относятся к одному и тому же образу, или не совпадают и относятся к различным образам. Используя сказанное, с помощью формулы (2) легко получаем, что исходная классификация изображений сохраняется в процессе самообучения, если для всех $N = 1, 2, \ldots$ будут выполняться неравенства

$$n^2(n-1)\nu_i + (n-1)^2\left(\sum_{j\in K_i}\nu_j - \sum_{j\in L_i}\nu_j\right) + |x_i| > 0 \quad (i = 1, 2, \ldots, 2n)$$

и тем более, если будут выполнены неравенства

$$n^2(n-1)\nu_i - (n-1)^2\sum_{j\in K_i\cup L_i}\nu_j + x > 0, \tag{4}$$

где через x обозначено минимальное из чисел $|x_i|$ ($i = 1, 2, \ldots, 2n$).

В свою очередь, нетрудно проверить, что неравенства (4) выполняются, если выполнены неравенства

$$\left|\frac{\nu_i}{N} - \frac{1}{2n}\right| < \frac{1}{4n^2}\left(1 + \frac{2x}{N(n-1)}\right) \quad (i = 1, 2, \ldots, 2n). \tag{5}$$

Величины $z_1 = \nu_1/N - \frac{1}{2n}$, $\ldots$, $z_{2n-1} = \nu_{2n-1}/N - \frac{1}{2n}$ распределены по закону, асимптотически стремящемуся к нормальному закону с плотностью вероятности вида

$$a = N^{n-1/2}e^{-NQ(z_1,\ldots,z_{2n-1})},$$

где a — некоторая положительная константа, а Q — положительно определенная квадратичная форма. Легко показать также, что неравенства (5) выполняются, если выполнены неравенства

$$\left|\frac{\nu_i}{N} - \frac{1}{2n}\right| < \frac{1}{8n^2} \quad (i = 1, 2, \ldots, 2n-1). \tag{6}$$

Для вероятности $q(N)$ неудовлетворения хотя бы одному из неравенств (6) получается оценка сверху вида

$$q(N) < cN^{n-1/2}\left(2\int_d^\infty e^{-bNx^2}dx\right)^{2n-1} <$$

$$< cN^{n-1/2}\left(-\frac{1}{bNd}\right)\int_d^\infty e^{-bNx^2}d(-bNx^2)^{2n-1} = \frac{f}{N^{n-1/2}}e^{-gN},$$

где c, d, f и q — положительные величины, не зависящие от N ($d = 1/8n^3$, $f = c/(bd)^{2n-1}$, $g = (2n-1)\,bd^2$). Константы c и b выбираются так, чтобы в результате использования формулы Муавра — Лапласа получить для вычисляемой вероятности приближение с избытком.

Вероятность выполнения хотя бы одного из неравенств (6) для значений N от M до ∞ не превосходит суммы ряда

$$\sum_{N=M}^{\infty} \frac{f}{N^{n-1/2}}\, e^{-gN},$$

а эта сумма, очевидно, меньше, чем

$$R(M) = \frac{f}{M^{n-1/2}} \cdot \frac{e^{-gM}}{1-e^{-g}}.$$

При $M \to \infty$ величина $R(M)$ стремится к нулю. Выберем M так, чтобы $R(M) < \varepsilon$.

Выбирая теперь $s = 2(M-1)\,n^2(n-1)$, получаем, что при $x > s$ неравенства (5) выполнены для всех значений $N = 1, 2, \ldots, M-1$. Вследствие же выбора M для всех остальных значений N неравенства (5) выполняются с вероятностью большей, чем $1 - \varepsilon$. Поскольку выполнение неравенства (5) для всех значений N от 1 до ∞ означает сохранение исходной классификации изображений, то теорема доказана.

Из теоремы 1 следует, что при достаточно больших начальных значениях выходных сигналов для всех изображений рассмотренный персептрон фактически почти лишен способности не только самообучаться, но даже просто изменять изначально задаваемую ему классификацию изображений. Из доказательства теоремы легко видеть, что остающаяся при этом слабая способность к самоизменению имеет наибольшую величину в случае правильной исходной классификации. Иными словами, персептрон наименее склонен к сохранению именно правильного способа функционирования.

Переходя к рассмотрению β-закона поощрения, фиксируем произвольное число β, заключенное между нулем и единицей, и рассматриваем произвольный симметричный персептрон C с β-законом поощрения, характеристическая матрица которого диагональна, т. е., иными словами, имеет отличные от нуля элементы только на главной диагонали. Как показано в [1], таким свойством обладает симметричный персептрон C_1 с (1, 1, 1)-нейронами [1], рассчитанный на распознавание горизонтальных и вертикальных линий, у которого входы каждого нейрона подсоединяются к элементам сетчатки, расположенным на одной горизонтали или на одной вертикали.

В случае β-закона поощрения основное рекуррентное соотношение для определения выходных сигналов можно записать в виде

$$V_i(l+1) = (1-\beta)\,(V_i(l) + T_{ij}\,\operatorname{sign} V_j(l)). \qquad (7)$$

Обозначения здесь такие же, как в формуле (2), причем это соотношение (как и в формуле (2)) справедливо для произвольных дискретных симметричных персептронов. В случае персептронов с диагональной характеристической матрицей оба слагаемых в правой части формул (2) и (7) всегда имеют один и тот же знак (случай, когда второе слагаемое равно нулю, исключаем из рассмотрения). Отсюда непосредственно вытекает справедливость следующего предложения.

Теорема 2. *Дискретный симметричный персептрон С с диагональной характеристической матрицей полностью лишен способности к самообучению (т. е. сохраняет неизменной любую задаваемую ему исходную классификацию изображений) как в случае α-закона поощрения, так и в случае β-закона поощрения.*

Легко видеть также, что выполняется и такое предложение.

Теорема 3. *Никакой дискретный симметричный персептрон (как с α-, так и с β-законом поощрения), работая в режиме самообучения, не может изменить исходную классификацию изображений, если эта классификация относит все изображения к одному и тому же образу.*

Нетрудно видеть, что полученные нами результаты можно рассматривать как контрпримеры к результатам Розенблатта [2] в той мере, в какой применяемые им рассуждения относятся не только к непрерывным, но и к дискретным нейронам. Во всяком случае, результаты данной статьи свидетельствуют о том, что асимптотическое поведение персептронов в режиме самообучения гораздо более сложно и требует значительно более тонких приемов для изучения по сравнению с приемами чисто качественного характера, используемыми Розенблаттом [2].

Для иллюстрации особенностей поведения персептронов в режиме самообучения по сравнению с режимом обучения рассмотрим случай, когда число изображений равно двум (каждый образ состоит из одного-единственного изображения). Этот случай допускает простую графическую интерпретацию.

Предварительно отметим, что в случае наличия двух образов (но при произвольном числе изображений) функционирование персептрона в режиме обучения так же, как и в режиме самообучения, удобно характеризовать вектором с компонентами $V_i(l)$ (см. выше). Основное рекуррентное соотношение для этих компонент будет иметь, очевидно, следующий вид:

$$V_i(lj) = V_i(l) \pm T_{ij}. \tag{8}$$

Это соотношение справедливо для любой пары изображений i, j и любой обучающей последовательности l. Второе слагаемое в правой части (8) выбирается со знаком плюс, если изображению j в правильной классификации соответствует положительный выходной сигнал, и со знаком минус, если соответствующий выходной сигнал должен быть отрицательным.

Рассмотрим дискретный симметричный персептрон с α-законом поощрения, характеристической матрицей которого является матрица $T = \begin{vmatrix} a, & b \\ b, & a \end{vmatrix}$, где $a > b > 0$.

Предположим, что при правильной классификации первое изображение должно индуцировать положительный выходной сигнал, а второе — отрицательный выходной сигнал. Откладывая по горизонтальной оси координату $V_1(l)$, а по вертикальной — $V_2(l)$, каждому вектору $V_1(l)$, $V_2(l)$ поставим в соответствие некоторую точку плоскости. Выбирая по одной точке в каждом квадранте, получаем наглядное представление действия формулы (8) (рис. 1).

На рис. 1 через T_1 обозначен вектор (a, b), а через T_2 — вектор (b, a). Характерной особенностью режима обучения является то, что направления векторов (определяющих случайные блуждания точки по решетке) не зависят от расположения точек на плоскости. Равнодействующая этих векторов всегда направлена в сторону того квадранта, в котором знаки

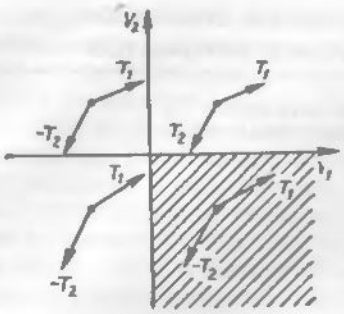


Рис. 1.

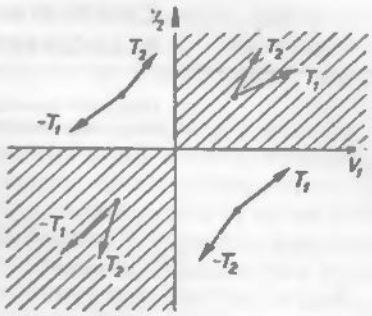


Рис. 2

координат (выходных сигналов персептрона) соответствуют правильной классификации изображений (в данном случае таким квадрантом является заштрихованный (четвертый) квадрант).

Для режима самообучения интерпретация соответствующей формулы (2) показана на рис. 2. В отличие от предыдущего случая направления векторов, определяющих случайное блуждание, различны в разных квадрантах (обозначения векторов такие же, как на рис. 1).

Видны качественные отличия ситуации, изображенной на рис. 2 и рис. 1. Прежде всего первый и третий квадранты (на рис. 2 они заштрихованы) здесь обладают ловушечным свойством: точка, попавшая в процессе случайного блуждания в один из этих квадрантов, уже не может никогда «выбраться» из него.

Попадание в эти квадранты фактически означает отсутствие какой-либо классификации (оба изображения причисляются к одному и тому же образу). Вместе с тем из квадранта, соответствующего правильной классификации (четвертый квадрант), как и из квадранта, соответствующего правильной классификации с точностью до знака выходного сигнала (третий квадрант), всегда существует ненулевая вероятность выхода в соседние квадранты.

Рассматривая возникшую ситуацию в чисто качественном плане, как это делает Розенблатт [2], мы должны были бы прийти к выводу, что изучаемый нами персептрон асимптотически стремится к состоянию, в котором на все изображения даются выходные сигналы одного и того же знака (отсутствие какой-либо классификации). Более точное рассмотрение (повторяющее выкладки, проведенные при доказательстве теоремы 1) приводит, однако, к совершенно иному выводу: как и в случае теоремы 1, при достаточном удалении начальной точки от границ квадранта вероятность продолжения случайного блуждания без выхода из этого квадранта во все последующие моменты времени (вплоть до бесконечности) может быть сделана сколь угодно близкой к единице.

Таким образом, еще раз подтверждается опасность, возникающая в случае, когда общие выводы об асимптотическом поведении персептронов в режиме самообучения основываются на рассуждениях чисто качественного характера, не подтвержденных точными расчетами и оценками.

СПИСОК ЛИТЕРАТУРЫ

1 *Глушков В. М.* Теория обучения одного класса дискретных персептронов // Журн. вычисл. математики и мат. физики.— 1962.— № 2.— С. 317—335.

2. *Rosenblatt F.* Two theorems of statistical separability in the perceptron. Symp. Mechaniz. Thought Proc., Teddington, England, 1958, Paper 1—3, 3—32.

САМООРГАНИЗУЮЩИЕСЯ СИСТЕМЫ И АБСТРАКТНАЯ ТЕОРИЯ АВТОМАТОВ

(Журнал вычислительной математики и математической физики.— 1962.— № 3)

Теорию дискретных самоорганизующихся систем, по-видимому, рационально строить на базе структурной теории автоматов (см. [1] или [2]). При этом самоорганизующаяся система представляется в виде логической сети, имеющей переменную структуру, а иногда и изменяющееся с течением времени число элементов. Оставаясь на уровне абстрактной теории автоматов (см. [2]), мы, естественно, проигрываем в богатстве возможных определений. Целью данной статьи является попытка показать, что, несмотря на указанные ограничения, и в рамках абстрактной теории автоматов можно строить достаточно содержательную теорию самоорганизующихся систем.

Абстрактным автоматом мы называем объект с конечным или счетным числом состояний $a_1, a_2, \ldots, a_n$, способный в каждый из моментов дискретного времени $t = 1, 2, \ldots, k, \ldots$, принимать входной сигнал $x(t)$ из некоторого конечного или счетного множества входных сигналов, переходить из предыдущего состояния $a(t-1)$ в следующее $a(t)$ и выдавать выходной сигнал $y(t)$ из некоторого конечного или счетного множества выходных сигналов. Соответствующие действия автомата задаются с помощью двух функций — функции переходов $a(t) = \delta[a(t-1), x(t)]$ и функции выходов $y(t) = \lambda[a(t-1), x(t)]$.

Фиксируя некоторое состояние a_0 в качестве начального состояния $a(t)$ автомата, превращаем его в преобразователь информации, преобразующий произвольные последовательности $x(1), x(2), \ldots, x(n)$ входных сигналов в соответствующие им последовательности $y(1), y(2), \ldots, y(n), \ldots$ выходных сигналов. При такой интерпретации действия автомата мы можем рассматривать его как преобразователь с неизменной структурой, лишенный каких-либо элементов самоорганизации.

Возможна, однако, и другая точка зрения, более соответствующая нашему обычному представлению о функционировании автоматов. Предположим, что тем или иным способом входная последовательность $x(1) \times x(2) \ldots x(n) \ldots$ разбита на отрезки $x(1)\,x(2) \ldots x(n_1)$, $x(n_1+1) \ldots x(n_2)$, $\ldots$, $x(n_{k-1}+1)\,x(n_{k-1}+2) \ldots x(n_k)$, $\ldots$, которые будем называть входными словами или вопросами и обозначать для краткости через $p_1, p_2, \ldots, p_{k-1} \ldots$ Соответствующие отрезки выходной последовательности

$$q_1 = y(1)\,y(2) \ldots y(n_1), \ \ldots, \ q_{k-1} = y(n_{k-1}+1)\,y(n_{k-1}+2) \ldots y(n_k)$$

(выходные слова) естественно называть ответами на соответствующие вопросы.

Ясно, что ответ автомата на тот или иной вопрос зависит теперь, вообще, не только от самого вопроса, но и от того места, которое данный вопрос занимает в рассматриваемой упорядоченной последовательности вопросов и ответов. Более точно, ответ на произвольный вопрос p_i нашей

последовательности зависит как от самого вопроса p_i, так и от последовательности предшествующих вопросов $p_1, \ldots, p_{i-1}$, составляющих (вместе с соответствующими им ответами $q_1, \ldots, q_{i-1}$) некоторую историю обучения рассматриваемого автомата.

При этом термины «вопрос» и «ответ» не должны пониматься слишком буквально. В частности, в каждый последующий вопрос, помимо собственно вопроса, может входить также оценка предыдущего ответа автомата. Не следует думать, что каждый ответ должен содержать то же число букв, что и соответствующий ему вопрос, поскольку с помощью введения как во входной, так и в выходной алфавиты автомата специальных пустых букв можно добиться любого соотношения между числом букв в любом данном вопросе и ответе. При этом собственно вопросом будет считаться вопрос с выброшенными из него пустыми буквами. Аналогичное положение наблюдается и в случае ответов.

Операцию разбиения входной и выходной информации на пары «вопрос — ответ» будем называть операцией циклирования. На практике обычно приходится иметь дело с двумя основными способами выполнения этой операции. Первый способ, называемый k-циклированием, заключается в том, что информация разбивается на отрезки длины k, так что все вопросы и ответы состоят из k букв (включая пустые буквы). Второй способ — это выделение конца одного вопроса и начала другого по некоторой фиксированной комбинации букв, называемой меткой (не исключено, что метка будет состоять из какой-либо одной, выделенной специально для этой цели, буквы).

При первом способе вопросы имеют одну и ту же длину, а общее число различных возможных вопросов в случае конечности входного алфавита непременно конечно. При втором способе циклирования вопросы могут иметь различную длину, а общее число вопросов может оказаться бесконечным, несмотря на конечность входного алфавита.

После того как тем или иным способом выполнено циклирование входной и выходной информации, можно осуществить так называемое циклическое приведение рассматриваемого автомата. С этой целью вводится новый входной алфавит, состоящий из всех вопросов в старом входном алфавите, рассматриваемых теперь как отдельные буквы. Аналогично буквами нового выходного алфавита явятся всевозможные ответы на указанные вопросы в старом выходном алфавите. Оставляя неизменным множество состояний рассматриваемого автомата A, можно построить новые функции переходов и выходов $\delta(a, p)$ и $\lambda(a, p)$, понимая под $\delta(a, p)$ состояние, в которое переходит автомат A из состояния a под действием входного слова (вопроса) p. Через $\lambda(a, p)$ обозначается ответ автомата A на вопрос p, если в качестве начального состояния выбирается состояние a. Ограничиваясь теперь лишь теми состояниями, в которые автомат A может переходить под действием всевозможных последовательностей вопросов, построим новый автомат B с определенными выше функциями переходов и выходов δ и λ. Этот автомат условимся называть циклическим приведением исходного автомата A.

После циклического приведения вопросы и ответы в автомате приобретают однобуквенную кодировку. Учитывая это, 1-циклированные автоматы (автоматы с однобуквенными вопросами и ответами) будем называть циклически приведенными, если они не имеют состояний, не достижимых из начального состояния.

Фиксируя способ циклирования входной и выходной информации, получаем возможность определять самоизменения в автоматах. Действи-

тельно, автомат, способный давать различные ответы на один и тот же вопрос (в зависимости от предшествовавших ему вопросов), естественно называть самоизменяющимся. Легко видеть, что справедливо следующее предложение.

Автомат тогда и только тогда является несамоизменяющимся, когда функция выходов $\lambda(a, p)$ его циклического приведения не зависит от состояния a.

Ясно, что в зависимости от характера циклирования входной и выходной информации один и тот же автомат может оказаться как самоизменяющимся, так и несамоизменяющимся. Рассмотрим, например, автомат A с двумя состояниями *1* и *2*, функции переходов и выходов которого задаются следующими таблицами:

	1	2
x	2	1
y	2	1

	1	2
x	u	v
y	v	u

При 1-циклировании этот автомат должен, очевидно, рассматриваться как самоизменяющийся. В случае же 2-циклирования после циклического приведения автомата A он преобразуется, как нетрудно видеть, в автомат с одним-единственным состоянием и поэтому должен рассматриваться как несамоизменяющийся автомат.

Возникает вопрос: можно ли определенное выше самоизменение отождествить с самоорганизацией? Опираясь на интуитивное представление о самоорганизации, мы должны называть самоорганизующимся такой автомат, который улучшает организацию своих ответов при улучшении организации возможных историй его обучения. Для количественной характеристики указанных улучшений естественно воспользоваться таким теоретико-вероятностным понятием, как энтропия. При этом целесообразно рассматривать две энтропийные характеристики, а именно энтропию обучения и энтропию ответов автомата.

Пусть $(p_1, p_2, \ldots, p_n) = P$ — последовательность вопросов (входных сигналов), заданных автомату в период его обучения. Эту последовательность будем называть обучающей. Предположим, что в той или иной фиксированной серии экспериментов с автоматом для каждой обучающей последовательности P задана вероятность $\rho(P)$ появления этой последовательности в экспериментах рассматриваемой серии (предполагается, что в пределах данной серии эта вероятность не меняется от эксперимента к эксперименту).

Тем самым задается некоторое распределение Q вероятностей $\rho(P)$ обучающих последовательностей. Энтропия этого распределения, которую будем называть энтропией обучения с данным законом распределения обучения, вычисляется по обычной формуле

$$H^Q(\text{обуч.}) = -\sum_P \rho(P) \log \rho(P). \qquad (1)$$

Для определенности условимся при подсчете энтропий пользоваться натуральными логарифмами.

В случае, когда фиксирован автомат A и его начальное состояние — a_0, всякое распределение вероятностей $\rho(P)$ обучающихся последовательностей однозначно определяет некоторое распределение вероятностей $\alpha(a)$ на множестве всех состояний этого автомата. Здесь через $\alpha(a)$ обозначена вероятность того, что после окончания процесса обучения автомата

он окажется в состоянии a. Если через S_a обозначить событие на входе автомата, представляемое состоянием a (множество входных слов, переводящих автомат из начального состояния в состояние a), то будет справедлива очевидная формула

$$\alpha(a) = \sum_{P \in S_a} \rho(P), \tag{2}$$

где суммирование распространено на все слова $p_1, p_2, \ldots, p_n$, содержащиеся в S_a (для краткости записи последовательность слов $P = (p_1, \ldots, p_n)$ отождествлена здесь со словом $p_1, p_2, \ldots, p_n$, составленным из элементов этой последовательности).

Зададимся некоторым распределением вероятностей $\gamma(p)$ вопросов p, задаваемых автомату *после* окончания процесса его обучения. Распределения $\alpha(a)$ и $\gamma(p)$ вместе с функциями переходов и выходов рассматриваемого автомата A однозначно определяют распределение вероятностей $\beta(p, q)$ для пар «вопрос (p) — ответ (q)». Энтропию этого последнего распределения будем называть энтропией экзамена с законом распределения обучения Q и обозначать через H^Q (экз.). Она определяется по формуле

$$H^Q(\text{экз.}) = -\Sigma\beta(p, q)\log\beta(p, q). \tag{3}$$

Используя в случае необходимости операцию циклического приведения автоматов, не нарушая общности, впоследствии сможем рассматривать лишь однобуквенные вопросы и ответы. При этом обучающие последовательности p превратятся в слова, составленные из отдельных составляющих их букв-вопросов, расположенных в том порядке, в котором они задавались автомату в процессе обучения.

Для дальнейшего построения теории необходимо задаться некоторым классом законов распределения обучения и присвоить каждому входящему в этот класс закону Q некоторую вероятность или, в случае непрерывных законов распределения, некоторую плотность вероятности $\rho(Q)$.

Простейший случай представляет схема независимых испытаний, когда на каждом шаге, как в режиме обучения, так и в режиме экзамена, вероятность $\gamma(P)$ появления любого данного вопроса постоянна и зависит только от этого вопроса. Ввиду ограничения лишь 1-циклированными автоматами задание закона распределения Q для обучения эквивалентно в этом случае присвоению некоторых вероятностей $\nu_i = \nu(x_1^i)$ появления на входе автомата каждой из букв x_i его входного алфавита. Сумма всех ν_i^1, разумеется, должна быть равной при этом единице.

Для схемы независимых испытаний естественно закон распределения для обучения отождествлять с вектором $\nu = (\nu_1, \nu_2, \ldots)$, составленным из вероятностей появления различных входных букв (входной алфавит предполагается при этом каким-либо образом упорядоченным). Класс законов наиболее естественно отождествлять с множеством всех векторов $\nu(\nu_1, \nu_2, \ldots)$, удовлетворяющих естественным ограничениям $0 \leqslant \nu_1 \leqslant 1$ и $\Sigma\nu_i = 1$ $(i = 1, 2, \ldots)$ с заданным на этом множестве равномерным законом распределения. Схему независимых испытаний с указанным выбором класса законов распределения условимся называть равномерной схемой независимых испытаний. При этом будем ограничиваться случаем, когда длина обучающей последовательности фиксирована, либо, в случае необходимости, предполагать, что эти длины описываются некоторым законом распределения (чаще всего пуассоновским).

Отметим, что в равномерной схеме независимых испытаний K выбор того или иного конкретного закона распределения определяет не только

вероятности тех или иных обучающих последовательностей, но и вероятности $\gamma(p)$ появления тех или иных вопросов на экзамене.

Это свойство целесообразно обобщить на произвольные классы K законов распределения, считая, что каждый элемент Q данного класса определяет пару законов распределения — как для обучающих последовательностей, так и для вопросов на экзамене. Будем предполагать также, что класс K выбирается так, чтобы он содержал один и только один закон распределения Q_0 с максимальной энтропией обучения H^{Q_0} (обуч.).

При этих предположениях нетрудно получить естественные количественные характеристики для способностей того или иного автомата к самоорганизации. Вводя приращения энтропий обучения и экзамена по формулам

$$\Delta H^Q(\text{обуч.}) = H^Q(\text{обуч.}) - H^{Q_0}(\text{обуч.}), \tag{4}$$

$$\Delta H^Q(\text{экз.}) = H^Q(\text{экз.}) - H^{Q_0}(\text{экз.}),$$

получаем возможность для любого автомата A и класса K законов распределения найти следующие две характеристики:

$$s(A, K) = -\int_K \Delta H^Q(\text{экз.})\, d\varphi(Q), \tag{5}$$

$$z(A, K) = \int_K \frac{\Delta H^Q(\text{экз.})}{\Delta H^Q(\text{обуч.})}\, d\varphi(Q). \tag{6}$$

Интегралы в этих формулах берутся по области, состоящей из всех законов рассматриваемого класса K. Чем больше величина этих интегралов, тем больше у рассматриваемого автомата A средняя способность к самоорганизации. Нулевому значению соответствует отсутствие способностей к самоорганизации, а отрицательные значения означают, что при улучшении организации обучения организация ответов автомата в среднем ухудшается. Иначе говоря, автомат ведет себя не как самоорганизующаяся, а как «самодезорганизующаяся» система.

Поскольку формула (6) приводит к значительно более сложным вычислениям, чем формула (5), в качестве основного количественного критерия для оценки способности автомата к самоорганизации будем выбирать критерий $s(A, K)$, а не критерий $z(A, K)$.

Пример. Рассмотрим два автомата A и B, функции переходов и выходов которых задаются следующими таблицами:

для автомата A

	1	2
x	1	1
y	2	2

,

	1	2
x	u	v
y	u	v

;

для автомата B

	1	2
x	1	2
y	2	2

,

	1	2
x	1	2
y	2	2

.

В этих таблицах цифрами 1 и 2 обозначены состояния автоматов, буквами x, y — входные сигналы (вопросы), а буквами u, v — выходные сигналы (ответы).

В качестве класса K законов распределения выберем такой класс, в котором вероятности появления вопросов x и y на экзамене равны друг другу, а законы распределения обучающих последовательностей возникают из схемы независимых испытаний при вероятностях появления вопросов x и y, равных p и $1-p$ соответственно (p пробегает в пределах класса K все значения от 0 до 1 с равными вероятностями). Кроме того, мы фиксируем длину n обучающих последовательностей, а соответствующий выбранному значению n критерий $s(A, K)$ будем обозначать через $s_n(A, K)$.

С учетом сделанных замечаний нетрудно вычислить значения критерия s_n для автоматов A и B:

$$s_n(A, K) = \int_0^1 \Big[p^2 \ln p^2 + 2p(1-p) \ln p(1-p) + (1-p)^2 \ln (1-p)^2 - \ln \frac{1}{4} \Big] dp = 2 \ln 2 + 2 \int_0^1 |p \ln p + (1-p) \ln (1-p)| \, dp = 2 \ln 2 - 1 \approx 0{,}38,$$

$$s_n(B, K) = \int_0^1 \Big[p^n \ln \frac{1}{2} p^n + (1-p^n) \ln \frac{1}{2}(1-p^n) - \frac{1}{2^n} \ln \frac{1}{2^{n+1}} \Big(1 - \frac{1}{2^n}\Big) \ln \frac{1}{2} \Big(1 - \frac{1}{2^n}\Big) \Big] dp =$$

$$= \frac{n \ln 2 + 1}{2^n} - \frac{n}{(n+1)^2} - \frac{1}{n} \Big[\frac{1}{1\Big(1+\frac{1}{n}\Big)\Big(2+\frac{1}{n}\Big)} + \frac{1}{2\Big(2+\frac{1}{n}\Big)\Big(3+\frac{1}{n}\Big)} + \cdots \Big] - \Big(\frac{1}{1 \cdot 2 \cdot 2^{2n}} + \frac{1}{2 \cdot 3 \cdot 2^{3n}} + \cdots \Big).$$

Используя последнее соотношение, находим следующую оценку:

$$s_n(B, K) < \frac{n \ln 2 + 1}{2^n} - \frac{n}{(n+1)^2} - \frac{1}{4n}.$$

Из этой оценки легко получить, что при $n \geqslant 5$ величина $s_n(B, K)$ отрицательна. Иными словами, при обучении последовательностями длины, большей 4, в выбранном классе законов распределения автомат B является в среднем «самодезорганизующимся», в то время как автомат A в тех же условиях обнаруживает способность к самоорганизации.

Отметим, что вывод о наличии или отсутствии у автомата способности к самоорганизации зависит от выбора класса законов распределения. Если, например, в рассмотренном примере в качестве K выбрать класс законов распределения, возникающий из равномерной схемы независимых испытаний, то, как нетрудно проверить, автомат B также стал бы в среднем самоорганизующимся, хотя величина этой самоорганизации оставалась бы меньшей, чем у автомата A.

При переходе от понятия самоорганизации к понятию самообучения уже нельзя удовольствоваться чисто теоретико-вероятностными понятиями. Необходимо вводить понятия, которые характеризовали бы ту или иную направленность процесса самоорганизации. С этой целью наиболее естественно ввести вещественную функцию $f(p, q)$, определенную на мно-

жестве всевозможных пар «вопрос (p) — ответ (q)», величина которой ха
рактеризует качество любого ответа q на любой данный вопрос p.

Как уже отмечалось выше, для любого данного автомата A с фикси
рованным начальным состоянием a_0 задание закона Q распределения ве
роятностей $\rho(P)$ на обучающих последовательностях P однозначн
определяет распределение вероятностей $\alpha(a)$ на множестве состояний ав
томата. Обозначим через $q = \lambda(a, p)$ ответ автомата A, приведенного пред
варительно в состояние a, на вопрос p, а через $\gamma(p)$ — вероятность появ
ления вопроса p. Величина

$$f^Q = \sum_{p,a} f[p, \lambda(a, p)]\,\gamma(p)\,\alpha(a)$$

представляет собой усредненный критерий качества ответов автомата н
«экзамене» при обучении его последовательностями, распределенными п
закону Q.

Теперь количеством самообучения автомата A естественно назват
разность $f^Q - f^{Q_0}$, где Q_0 — априорный закон распределения вероятнос
тей обучающих последовательностей, известный конструктору в момен
создания автомата, а Q — апостериорный закон распределения, которы
фактически имел место для некоторого класса экспериментов по обучению
Как правило, энтропия распределения Q меньше, чем энтропия распреде
ления Q_0.

Если задан класс K апостериорных законов распределения Q с плот
ностью вероятности $\varphi(Q)$, то интеграл

$$s_f(A, K) = \int_{Q \in K} (f^Q - f^{Q_0})\,\varphi(Q)\,dQ$$

будет представлять собой усредненную количественную характеристику
для способности рассматриваемого автомата к самообучению (применш
тельно к выбранному классу K, автомату A и вещественной функции f).

Подобно тому как это имело место в случае самоорганизации, наряду
с критерием $s_f(A, K)$ иногда целесообразно рассматривать также критери

$$z_f(A, K) = -\int_{Q \in K} \frac{f^Q - f^{Q_0}}{H(Q) - H(Q_0)}\,\varphi(Q)\,dQ,$$

где через $H(Q)$ и $H(Q_0)$ обозначены энтропии законов распределения Q
и Q_0.

СПИСОК ЛИТЕРАТУРЫ

1. *Глушков В. М.* Некоторые проблемы синтеза цифровых автоматов // Журн
вычисл. математики и мат. физики.— 1961.— 1, № 3.— С. 371—411.

2. *Глушков В. М.* Абстрактная теория автоматов // Успехи мат. наук.— 1961,—
16, вып. 5.— С. 3—62.

НЕКОТОРЫЕ ПРОБЛЕМЫ ТЕОРИИ АВТОМАТОВ И ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

(Кибернетика.— 1970.— № 2)

Как и всякая наука, в начале своего развития теория автоматов черпала новые постановки задач прежде всего из запросов практики. В настоящее время основной областью практического приложения теории автоматов является проектирование электронных цифровых машин. Однако публикации последних лет обнаруживают тенденцию теории автоматов к развитию в основном именно тех задач теории автоматов, которые находятся в отрыве от практики. В результате этого, например, среди части американских инженеров, проектирующих ЭЦМ, складывается отношение к теории автоматов как к «академической», оторванной от жизни науке.

Разумеется, любая наука имеет право на собственные внутренние постановки задач и обобщения, диктуемые требованиями развития аппарата самой науки. Более того, многие разделы математики в период своей зрелости успешно развивались и продолжают развиваться преимущественно таким способом. Успех этого развития базируется на прочном фундаменте практических приложений. Для новой же области науки, какой все еще остается теория автоматов, подобный отрыв теории от практики связан с большими опасностями.

Каковы же основные проблемы современной теории автоматов, диктуемые проблемами практики?

Рассмотрим некоторые из таких проблем, развиваемых киевской школой теории автоматов.

Как известно, теория конечных автоматов, по крайней мере в первый период развития, исходила из практических задач синтеза и оптимизации схем ЭЦМ. На базе полученных результатов в Институте кибернетики АН УССР была построена так называемая малая система автоматизации проектирования ЭЦМ, которая успешно эксплуатируется в ряде организаций не только в Киеве, но и в Москве. Эта система состоит из нескольких десятков взаимосвязанных программ общим объемом около 30 тыс. команд. Она позволяет автоматически проектировать схемы дискретных автоматов, сложность которых не превосходит 10^7 (под сложностью автомата здесь понимается произведение числа различных входных сигналов на число внутренних состояний автомата).

Входной информацией для системы является задаваемое тем или иным способом отображение, индуцируемое искомым автоматом. На выходе системы получается функциональная схема проектируемого автомата, построенная из заданных радиотехнических элементов с учетом простейших требований надежности. Набор элементов, о которых идет речь, может варьироваться в довольно широких пределах. При этом, в отличие от обычных абстрактных постановок, в набор включаются различного рода усилители и формирователи, обеспечивающие восстановление необходимого уровня сигнала или его формы. В состав системы входят программы для синтеза и минимизации абстрактного автомата, для кодирования состояний автомата, для получения функций возбуждения к (приближенной) минимизации их в заданной системе элементов.

Система реализована на ЭЦМ класса «М-20» с быстродействием в несколько десятков тысяч операций в секунду. Однако даже при повышении быстродействия машины в тысячи раз вряд ли на подобном пути удастся превзойти порог сложности автоматов 10^9. Подобная сложность приемлема при синтезе устройств управления современных ЭЦМ. В то же время арифметические устройства даже относительно простых параллельных ЭЦМ имеют сложность порядка 10^{30} и выше.

Оставаясь в рамках сложившихся методов теории конечных автоматов, вряд ли можно решить задачу полной формализации и автоматизации синтеза ЭЦМ. Для применения этой теории необходимо предварительно разбить схему проектируемой машины на достаточно мелкие блоки. Подобное разбиение можно выполнить лишь после того, как выбраны основные алгоритмы функционирования машины и составлены микропрограммы этих алгоритмов. Этот так называемый алгоритмический этап проектирования ЭЦМ до последнего времени был совершенно не формализован и выполнялся исключительно на основании интуиции и опыта проектировщиков.

Для формализации алгоритмического этапа понятия конечного автомата недостаточно. Вместе с тем существующие в настоящее время концепции бесконечных и так называемых растущих автоматов ориентируются в основном на теоретические алгоритмические системы (машины Тьюринга) и не приемлемые для решения практических задач проектирования ЭЦМ. Несколько ближе к практическим запросам понятие магазинного автомата, но и здесь направление ведущихся разработок далеко от непосредственных запросов практики. Неудивительно, что в результате развитие теории бесконечных автоматов идет само по себе, в то время как естественная сфера приложения этой теории — алгоритмический синтез ЭЦМ,— находится в тисках эмпирики.

Путь исправления создавшегося положения в принципе ничем не отличается от пути, ведущего к превращению любой эмпирической науки в дедуктивную. В чем же состоят основные этапы этого пути? Первый этап — это этап создания формального языка для описания понятий и процессов, изучаемых данной наукой.

При этом существенно, чтобы форма языка наиболее полно соответствовала кругу изучаемых понятий. Хотя, в принципе, язык алгоритмов Поста или машины Тьюринга достаточен для описания любых вычислительных процедур, никому не придет в голову использовать эти языки для программирования ЭЦМ. Для этих целей созданы и создаются специальные языки (АЛГОЛ, ФОРТРАН и др.). В нашем случае необходимо иметь формальный язык, ориентированный на описание алгоритмов функционирования электронных цифровых машин и их отдельных устройств.

Однако для превращения эмпирической науки в дедуктивную недостаточно одного лишь формального языка. Необходимо иметь алгебру языка, позволяющую производить эквивалентные преобразования выражений в этом языке, например современный язык алгебры и анализа был бы бесполезен для развития математики, если бы не было средств для преобразования выражения $(a + b)^2$ в выражение $a^2 + 2ab + b^2$; выражения $\int_0^a x dx$ в $\frac{1}{2} a^2$ и т. п. Разработка алгебры формального языка той или иной науки и составляет содержание второго этапа превращения из эмпирической науки в дедуктивную.

Наконец, развитая алгебра языка позволяет ставить и решать проблемы оптимизации, т. е. нахождения наилучших в том или ином смысле форм

представления выражений в языке. Этот третий этап тесно связан со вторым, часто переплетаясь с ним по времени. Критерии оптимизации могут быть не только точными количественными, но и качественными, условными. Например, в анализе выражение функции через квадратуры считается обычно более приемлемым, чем представление ее дифференциальным уравнением с теми или иными начальными условиями.

Современная прикладная теория алгоритмов находится в начале первого этапа своего развития. Оформилось некоторое количество формальных алгоритмических языков (АЛГОЛ, КОБОЛ, ЛИСП и др.), ориентированных на те или иные классы прикладных задач. Однако развитие этих языков нельзя считать законченным. Действительно, хороший формальный язык должен обладать развитой системой сокращенных обозначений для упрощения записей часто встречающихся выражений. В то же время в таком, например, языке как АЛГОЛ используются в основном лишь те сокращенные обозначения (да и то не все), которые имеются в языке алгебры и анализа, сложившемся задолго до создания АЛГОЛА, и включены в АЛГОЛ практически без изменений. Речь идет о языке алгебраических формул и сокращенных обозначениях для простейших элементарных функций. В то же время такие элементы общих алгоритмических языков, как циклы и условные переходы, остались в АЛГОЛЕ фактически на уровне простых словесных описаний.

Развитый алгоритмический язык должен включать в себя богатую систему стандартных процедур для операций с массивами так, чтобы значительная часть алгоритмов, содержащих циклы и переключения, могла бы описываться с помощью этих процедур без явного использования операторов цикла и условных переходов. Таким образом, алгоритмическим языкам предстоит еще пройти немалый путь, прежде чем они станут по степени разработанности на один уровень с языком аналитических выражений, являющимся по сути не более чем фрагментом этих языков.

Разумеется, для начала работы по алгебре языка нет необходимости ждать полного завершения работы по окончательной шлифовке самого языка. Именно так шло развитие языка аналитических выражений. Однако, что касается общих алгоритмических языков, то до самого последнего времени не было по сути даже попыток создания алгебры языка. Имеющиеся в небольшом количестве работы по формальным преобразованиям схем программ не могут идти в счет, так как они не меняют последовательности элементарных операций, из которых складывается выполнение алгоритма, т. е. оперируют весьма небольшой частью эквивалентных преобразований алгоритмов. Возможно, исследователей отпугивало то, что, как известно, проблема эквивалентности алгоритмов алгоритмически неразрешима.

Вместе с тем задача автоматизации синтеза ЭЦМ настоятельно требует развития алгебры эквивалентных преобразований алгоритмов, наличие которой позволяет решить задачу автоматизации ЭЦМ в следующем смысле: задавая комплекс алгоритмов, которые должна реализовать проектируемая машина тем или иным образом (например, в соответствии с их определениями), преобразовывать их формальным образом в другие формы, лучше соответствующие тем задачам, которые должна решать машина.

При этом записи алгоритмов должны быть выполнены на таком уровне, чтобы обеспечить естественный переход к последующей их реализации в виде системы взаимодействующих автоматов.

Таким образом, с точки зрения задач синтеза ЭЦМ одним из основных вопросов современной прикладной теории автоматов является разработка

автоматно-алгоритмического языка, ориентированного на описание структур ЭЦМ, и развитие алгебры этого языка.

Развитие алгебры алгоритмических языков, т. е. техники эквивалентных преобразований выражений в этих языках, имеет и другое, общематематическое и даже философское значение. Хорошо известно, что сегодня в прикладной математике резко различаются аналитические и численные методы решения задач. Среди части аналитиков господствует пренебрежительное отношение к численным методам, а некоторые математики вообще не склонны причислять к математике методы решения прикладных задач с помощью программ для ЭЦМ.

В чем причина такого неравноправного положения численных методов? Быть может, в меньшей степени общности даваемых ими решений? Разумеется, нет. Хорошо известно, что разумно составленная типовая программа обеспечивает обычно бо́льшую степень общности решения, чем аналитические методы, применение которых связано, как правило, со всевозможными упрощениями в постановке задач.

Действительное преимущество аналитических решений по сравнению с численными, алгоритмическими методами состоит в развитости языка аналитических выражений и алгебры этого языка. Благодаря этому аналитические выражения обычно гораздо компактнее по сравнению с программами. Из-за хорошей изученности элементов, с помощью которых строятся аналитические выражения, относительно легко угадывается качественное поведение решений, их зависимость от параметров. Наконец, развитая алгебра языка позволяет за одним выражением видеть целое семейство эквивалентных ему выражений. Соотношения типа

$$(a+b)(a-b)=a^2-b^2,\ d\sin x=\cos x dx,\ \int x^2dx=\frac{1}{3}x^2+c$$

и другие за несколько столетий развития языка аналитических выражений вошли в плоть и кровь не только математиков-профессионалов, но и всех тех, кто применяет математику.

Нетрудно понять, что развитие общих алгоритмических языков и алгебры таких языков приведет к тому, что выражения в этих языках (сегодняшние программы для ЭЦМ) станут столь же привычными, понятными и удобными, какими сегодня являются аналитические выражения. При этом фактически исчезает различие между аналитическими и общими алгоритмическими методами, что приведет к коренному изменению облика самой математики. С позиций этой новой математики те, кто сегодня смотрит свысока на программистские методы, будут выглядеть примерно так же, как сейчас выглядят некоторые математики прошлого века, упорно считавшие квинтэссенцией математики нахождение методов решения в квадратурах тех или иных классов дифференциальных уравнений.

Подобное изменение облика математики будет иметь колоссальное значение для развития всех наук, в том числе тех, которые сегодня почти не пользуются математикой. Дело в том, что язык аналитических выражений складывался под влиянием лишь небольшой части наук, в первую очередь механики и астрономии, а затем физики и в меньшей мере химии. Неудивительно, что описывая достаточно хорошо процессы, изучаемые этими науками, язык аналитических выражений оказывается бессильным для описания качественно отличных от них сложных процессов в экономике, в живых организмах, в эволюции, в лингвистике и т. п. Бессмысленно, например, искать формулу перевода с одного языка на другой. Что же касается алгоритмов перевода, то они создаются и совершенствуются.

«Уравнивание в правах» алгоритмических методов с аналитическими

приведет к огромному расширению сферы приложений математики, к резкому увеличению темпов математизации наук, которые ныне считаются чисто экспериментальными или описательными.

Разумеется, задача, о которой идет речь, огромна и требует для своего полного решения сотрудничества многих коллективов и не одного поколения математиков. Однако ряд шагов в направлении ее решения можно сделать уже сегодня. Начало созданию алгебры алгоритмов было положено в работах, выполненных в Институте кибернетики АН УССР. Эти работы имеют своей непосредственной целью создание математической основы для автоматизации алгоритмического этапа проектирования ЭЦМ. Однако принципы, положенные в их основу, могут использоваться и для решения гораздо более общих задач.

Основная идея этих работ состоит в представлении любого алгоритма в виде схемы взаимодействия двух автоматов, а реализуемого алгоритмом отображения — в виде элемента некоторой алгебры.

Рассмотрим пару автоматов A и B, первый из которых представляет собой конечный автомат Мили, а второй — автомат Мура (вообще говоря, бесконечный). Автомат A, который мы будем называть управляющим, служит для представления алгоритма, а автомат B, называемый операционным, представляет перерабатываемую алгоритмом информацию. Его состояния отождествляются с преобразуемыми алгоритмом объектами (словами, множествами слов, числами и т. п.).

Преобразованием или оператором называется отображение (вообще говоря, лишь частично определенное) множества M этих объектов (так называемого информационного множества) в себя. Кроме операторов на множестве M определены условия, истинные для одних элементов M, ложные для других и неопределенные для третьих. Некоторые операторы и условия объявляются элементарными. Задача теории алгоритмов состоит в указании способов представления сложных операторов через элементарные операторы и элементарные условия.

Один из этих способов — содержательный или автоматный — задается схемой взаимодействия двух автоматов (см. рисунок).

Выходные сигналы y_i управляющего автомата A отождествляются в этой схеме с элементарными операторами, осуществляющими (как входные сигналы автомата B) соответствующие преобразования множества состояний автомата B или, что то же,— информационного множества M. Входными сигналами x_i автомата A являются генерируемые автоматом B (конечные) кортежи элементарных условий. При желании они могут рассматриваться как буквы некоторого абстрактного алфавита.

Кроме уже описанных элементов схемы одно из состояний автомата A объявляется начальным, а другое — заключительным. Содержательный способ представления алгоритма состоит в мысленном воспроизведении схемы взаимодействия автоматов A и B так, чтобы взаимодействие начиналось, когда автомат A находится в начальном состоянии, а заканчивалось, когда он в первый раз перейдет в заключительное состояние.

При фиксированных множествах элементарных операторов и условий реализуемый схемой алгоритм полностью задается выбором управляющего автомата A вместе с его начальным и заключительным состояниями. Соответствующий этому алгоритму оператор (отображение множества M в себя) может быть одним и тем же для различных автоматов A, которые в этом случае называются слабо эквивалентными относительно автомата B. При сильной эквивалентности относительно B дополнительно тре-

буется, чтобы при любом данном начальном состоянии B последовательности элементарных операторов, выданных сравниваемыми управляющими автоматами, до их остановки в заключительных состояниях были одинаковыми.

Преобразования схем программ, изучавшиеся Яновым и другими авторами, относятся к классу сильно эквивалентных преобразований. Предлагаемая техника сводит вопрос о сильной эквивалентности к обычной классической эквивалентности частично определенных автоматов. Это позволяет усилить результаты Янова и одновременно значительно упростить доказательства.

Для эффективного выполнения слабо эквивалентных преобразований строится пара алгебр $\mathfrak{A}$ и $\mathfrak{B}$. Элементами первой алгебры являются операторы на заданном информационном множестве M, элементами алгебры $\mathfrak{B}$ — условия, определенные на M.

В множество операций алгебры $\mathfrak{A}$ входит обычное умножение операторов. Кроме того, для каждого условия α из алгебры $\mathfrak{B}$ в алгебре $\mathfrak{A}$ определяется одна бинарная и одна унарная операции. Бинарная операция называется α-дизъюнкцией. Она переводит пару операторов P, Q в новый оператор $R = (P \underset{\alpha}{\vee} Q)$, совпадающий с оператором P на множестве элементов M, для которых условие α истинно, и с оператором Q на множестве тех элементов, для которых оно ложно. Эта операция представляет собой формализацию хорошо известного понятия кусочного задания операторов.

Унарная операция носит название α-итерации. Она оператор S сопоставляет с новым оператором $T = \{S\}_{\alpha}$, который получается в результате применения оператора S такое число раз (быть может, равное нулю), чтобы условие α стало истинным (в случае бесконечного повторения оператора S оператор T считается неопределенным).

В алгебре условий $\mathfrak{B}$ кроме обычных операций дизъюнкции, конъюнкции и отрицания условий вводится операция (левого) умножения на элементы алгебры $\mathfrak{A}$. Произведение $P \cdot \alpha$ представляет собой условие, истинное или ложное соответственно тому, будет ли истинным или ложным после выполнения оператора P условие α.

Алгебра алгоритмов с данными множествами элементарных операторов C и элементарных условий D — это минимальная пара алгебр $(\mathfrak{A}, \mathfrak{B})$ со введенными выше операциями, порожденная множествами C и D. Каждый оператор P из алгебры $\mathfrak{A}$ можно представить в виде формулы, выражающей его с помощью введенных выше операций через элементарные операторы и элементарные условия. Эта формула носит название регулярной программы данного оператора P.

Основная теорема теории состоит в возможности приведения к регулярной форме любой программы (алгоритма), пользующейся этой же системой элементарных операторов и элементарных условий. Будучи записан в такой форме, алгоритм становится элементом алгебры алгоритмов $\mathfrak{A}$, выраженным через образующие элементы этой алгебры. Выписав систему определяющих соотношений алгебры $\mathfrak{A}$, можно осуществлять преобразования регулярных программ этого алгоритма, находя наилучшие с той или иной точки зрения программы.

Пользуясь такой методикой, легко решать, например, такие задачи, как преобразования алгоритмов умножения и деления целых чисел, записанных в соответствии с их определениями, в формы, обычно используемые в позиционных системах счисления. Разумеется, для алгоритмов такого рода, формы представления которых шлифовались в течение мно-

гих столетий, применение техники формальных преобразований является не более чем простой иллюстрацией возможностей этой техники. Однако, когда мы переходим к преобразованию сложных неарифметических алгоритмов, преимущества, задаваемые подобной техникой, могут стать решающими: вместо случайных, эмпирических поисков лучших представлений алгоритмов появляются четкие правила преобразований, позволяющие формализовать и автоматизировать оптимизацию алгоритмов. При этом вследствие простоты перехода от регулярной записи к автоматной (содержательной) форме представления алгоритмов относительно просто решается задача инженерного синтеза автомата (или системы автоматов), реализующих данную алгоритмическую структуру проектируемой ЭЦМ.

Возникает возможность построить такую систему программ, которая во взаимодействии с конструктором позволяет от описания (в соответствии с определениями) алгоритмов, реализуемых проектируемой ЭЦМ, перейти к функциональным и даже монтажным схемам этой ЭЦМ. Подобная система, называемая большой системой автоматизации проектирования ЭЦМ, разрабатывается в настоящее время в Институте кибернетики АН УССР. Эта система включает в себя трансляторы для преобразования записей в формальных языках, описывающих проектируемую машину на различных уровнях (от алгоритмического до монтажного). Кроме того, в систему включаются программы для оценки качества проекта на различных стадиях по различным критериям (например, математического ожидания скорости выполнения той или иной операции).

Однако центральную роль должны играть здесь программы оптимизации. На современном уровне развития теории полностью автоматическую систему оптимизации построить не удается. Поэтому в большой системе автоматизации проектирования предусматриваются средства, обеспечивающие совместную работу системы и конструктора. Взаимодействие между ними состоит в том, что конструктор указывает пути преобразования формальных выражений, описывающих проектируемую ЭЦМ или ее отдельные части, а машинная система выполняет перебор возможностей в рамках намеченного пути, осуществляет выбор лучшей схемы и выдает конструктору оценку ее качества, после чего процесс может повторяться.

Весьма существенным является вопрос о выборе системы элементарных операторов и элементарных условий, на которых зиждется построение используемой в системе алгебры алгоритмов. Для принятых в современных ЭЦМ параллельных методов обработки информации на длинных регистрах оказывается возможным построить алгоритмы, которые автоматически выдают последовательности элементарных операторов и элементарных условий в порядке увеличения сложности их реализации.

Основная идея состоит во введении так называемых периодически определенных преобразований на регистрах. В простейшем случае одного двоичного регистра эту идею можно объяснить следующим образом. Пусть x_i — булева переменная, значения которой выражают содержание i-ячейки рассматриваемого регистра. Будем считать для простоты, что i изменяется в интервале $(-\infty, \infty)$. При выполнении преобразования на регистре новое значение x_i переменной x_i определится как некоторая (булева) функция от конечного числа переменных x_j. В частности, для $i = 0$ имеем $x_0 = f(x_{i_1}, \ldots, x_{i_k})$. Если для любого i преобразование выражается той же функцией f, но с соответственно сдвинутыми индексами переменных, т. е. $x' = f(x_{i_1+i}, \ldots, x_{i_k+i})$, то соответствующее преобразование на регистре будет называться периодически определенным, а функция f его производящей функцией.

Сложность схемной реализации этого преобразования зависит от сложности P булевой функции f (например, от общего количества операций в ее минимальной ДНФ) и длины Q соединительных проводов, определяемых величиной $\max |i_j|$ ($j = 1, 2, \ldots, k$). Некоторая функция F от величин P и Q и будет определять сложность реализации рассматриваемого элементарного преобразования. Вид этой функции зависит от уровня технологии. Например, для машин первого и второго поколения с навесными элементами эта функция слабо зависела от Q, в то же время для микроэлектронной технологии у машин третьего и четвертого поколений влияние параметров P и Q начинает уравниваться.

Как бы то ни было, зафиксировав функцию F, мы может легко построить алгоритм, выдающий последовательность периодически определенных преобразований на регистре в порядке возрастания сложности их реализации. Например, если $Q = \max |i_j|$, P — количество операций в минимвльной ДНФ, а $F(P, Q) = P + Q$, то к числу операций сложности 1 относятся инверсия на регистре, а также правый и левый сдвиг на одну позицию. Все обычно используемые инженерами элементарные операции относятся к числу минимальных по сложности реализации и выдаются соответствующим алгоритмом на самых первых шагах.

Следует лишь иметь в виду, что приведенное определение периодически определенных преобразований относится к самому простому случаю. В общем случае регистров может быть несколько, могут существовать также фиктивные регистры, представляющие цепи, аналогичные цепям переноса при сложении. Однако общая идея остается прежней. С небольшими изменениями она работает также для случая генерации последовательности элементарных условий.

Отметим, наконец, что некоторые типы определяющих соотношений в алгебре алгоритмов могут также находиться автоматически. Речь идет об определяющих соотношениях типа $p_1p_2\ldots p_m = q_1q_2\ldots q_n$ и $r_1r_2, \ldots, r_k\alpha = \varphi(\alpha_1, \alpha_2, \ldots, \alpha_l)$, где p_i, q_j, r_s — элементарные операторы, α_v — элементарные условия, а φ — произвольная булева функция. Основная идея здесь заключается в том, что если элементарные операторы заданы как периодически определенные преобразования, то их произведение будет также периодически определенным, причем производящая функция этого произведения будет вычисляться по строго определенным правилам через производящие функции сомножителей. Аналогичное положение наблюдается для произведения операторов на условия и для булевых функций от условий. Таким образом, нахождение соотношений указанных двух типов сводится к проверке тождественности двух булевых функций, что, как известно, в принципе решается просто.

Алгоритм для автоматического нахождения определяющих соотношений осуществляет последовательный (в порядке возрастания суммарной длины слов в левой и правой частях) перебор всех равенств указанных выше двух типов и проверяет истинность этих равенств, оставляя истинные равенства в качестве искомых соотношений. Разумеется, простым перебором, без соответствующей эвристики, нельзя получить слишком сложные соотношения. Тем не менее соотношения вида $s^2l = ls$, важные для уяснения выше преобразований алгоритмов умножения и деления подобным образом получить можно. Сложность этих соотношений, т. е. сумма длин слов в левой и правой частях, равна 5, перебор же начинается со сложности 3 (равенства вида $pq = r$).

Более сложные соотношения в алгебре алгоритмов, затрагивающие операции α-дизъюнкции и α-итерации, требуют для своего вывода боле

сложную технику. Опыт, однако, показывает, что для практически эффективной работы в алгебре алгоритмов оказывается достаточным пользоваться лишь тождественными соотношениями подобного типа. Иными словами, эти соотношения могут быть установлены раз и навсегда, независимо от выбора системы элементарных операторов и элементарных условий.

Переходя к проблемам искусственного интеллекта, прежде всего отметим, что в отличие от теории автоматов, здесь нет единой практической задачи, определяющей развитие теории, хотя связь с практикой и тут самая тесная. Вместе с тем, как следует из самого названия, исследования по искусственному интелекту имеют единую конечную принципиальную цель — создать модель человеческого мозга и разгадать тайны мышления. Что же касается частных проблем, то здесь их великое множество: от эвристических программ игры в шахматы до создания физической модели нейрона, от общей теории самоорганизующихся систем до конкретных алгоритмов и устройств для распознавания зрительных или звуковых образов.

Имея в виду направление киевской школы теоретической кибернетики, остановимся подробнее на двух проблемах. Первая — это проблема обучения языку. Как известно, в Институте кибернетики АН УССР были построены и успешно работают программы по обучению смыслу фраз в естественных языках. Более точно — эти программы позволяют достаточно эффективным способом расширять заданные множества осмысленных и бессмысленных фраз, используя некоторый фиксированный заранее словарь и некоторые ограничения на грамматическую структуру фраз.

Эти исследования позволяют сформулировать общую проблему обучения языку в следующем виде.

Будем рассматривать естественные языки, в качестве букв которых используются отдельные грамматические элементы слов: корни, префиксы, суффиксы, окончания. Фиксируем класс формальных грамматик для описания не только обычной грамматики, но и семантики языков. В качестве такого класса по ряду соображений удобно выбрать класс контекстно свободных грамматик. Описание формальной грамматической семантики языка понимается в том смысле, что эта грамматика должна порождать не только грамматически правильные, но и осмысленные фразы языка. Поскольку формальное понятие осмысленности до построения соответствующей грамматики отсутствует, для проверки осмысленности можно использовать тот или иной неформальный прием, например голосование фиксированной заранее группы экспертов.

Проблема, о которой идет речь, заключается в построении алгоритма, способного решить следующую задачу: по любому заданному конечному набору M осмысленных фраз, и, быть может, по некоторому набору бессмысленных фраз построить контекстно свободную грамматику G, которая: а) продолжает все фразы из M; б) не порождает ни одной бессмысленной фразы; в) имеет наименьшую сложность записи из всех грамматик, удовлетворяющих двум предыдущим условиям.

Сложность записи грамматики можно отождествить с суммарным числом букв во всех ее продукциях или с числом самих продукций. Экстраполирующей силой (по отношению к M) построенной грамматики будем называть логарифм отношения общего числа порождаемых ею осмысленных фраз к числу фраз в исходном наборе M. Чтобы это отношение было конечным, можно заранее ограничить максимальную длину генерируемых грамматикой фраз.

Нетрудно понять, что поставленная задача всегда имеет решение, независимо от того, является ли рассматриваемый язык контекстно свободным или нет. Действительно, вводя одну единственную терминальную букву x, всегда можно породить конечное множество M фраз (и только эти фразы) грамматикой, состоящей из продукции вида $x \to p$ ($p \in M$). Ясно, что эта грамматика удовлетворяет условиям «а» и «б» приведенного выше определения. Подобное тривиальное решение вопроса о порождении фраз множества M имеет наиболее громоздкую запись грамматики, а экстраполирующая сила этой грамматики равна, очевидно, нулю.

Полное решение поставленной задачи дает одна из конечного числа грамматик, сложность записи которой не превосходит сложности записи построенной тривиальной грамматики. Следовательно, путем перебора мы можем в принципе решить поставленную задачу. Однако само собой разумеется, что алгоритм, основанный на полном переборе грамматик, непригоден для практического применения. Поэтому проблема состоит в построении достаточно экономичного алгоритма для нахождения требуемой грамматики.

Опыт показывает, что, как правило, при уменьшении сложности записи искомой грамматики увеличивается ее экстраполирующая сила. Ясно также, что способность по фрагменту языка восстановить описывающую этот фрагмент грамматику достаточно большой экстраполирующей силы свидетельствует о степени «интеллектуальности» восстанавливающего алгоритма. Сама же экстраполирующая сила может служить количественной мерой степени интеллектуальности объекта (машины или живого существа), реализующего этот алгоритм. Таким образом, поставленная выше задача — это типичная задача «искусственного интеллекта», а ее решение позволило бы достаточно эффективно обучать машины человеческим языкам. Попутно, имея оценку для высшего уровня языковой «интеллектуальности» (экстраполирующей), мы получили бы своеобразную шкалу для сравнения интеллектов животных и различных алгоритмов для обучения машин языку. Понятно, что поскольку экстраполирующая сила грамматики (и сама грамматика) меняется при изменении фрагмента языка, необходимо при ее использовании в качестве меры интеллектуальности проводить то или иное усреднение по различным фрагментам.

Использование контекстно свободных грамматик приводит к тому, что задача восстановления грамматики с достаточно экономной записью требует образования абстрактных понятий (например, понятия одушевленного или неодушевленного предмета). Это существенно более сложный процесс, чем процесс выработки условных рефлексов, хотя и этот последний процесс можно описать в терминах восстановления некоторой грамматики в языке пар «стимул — реакция». При этом для выработки любого условного рефлекса достаточно уметь восстанавливать описанную выше тривиальную грамматику.

Использование языков пар позволит в будущем решать задачу установления связи между чувственными образами (классифицируемыми устройствами типа персептронов) и их словесными описаниями. Нетрудно понять, что построенные таким образом модели будут охватывать весьма весомый участок деятельности мозга человека.

Что же касается логического мышления, то, как нам кажется, хорошей (если не лучшей) модельной задачей здесь может служить задача автоматизации доказательств теорем в математике. Вряд ли стоит специально подчеркивать, что эта задача имеет и немалое прикладное значение.

так как ее достаточно эффективное решение позволит существенно ускорить прогресс математики и всех дедуктивных наук.

При автоматизации доказательства теорем как раз и составляют суть второй проблемы из области искусственного интеллекта, которую мы намеревались осветить. Анализируя работы, выполненные в этом направлении (в том числе и в Институте кибернетики АН УССР), нельзя не обратить внимания на один присущий всем им недостаток. Все эти работы имеют своей целью создать универсальную программу, способную самостоятельно, без участия человека доказать если не все теоремы вообще, то во всяком случае — все предложения в достаточно широкой области математики.

Такая постановка вопроса не соответствует опыту, накопленному в других областях применения ЭЦМ. Действительно, вряд ли кто-нибудь сомневается, что доказательство теорем представляет собой в общем более трудную и, самое главное, менее изученную проблему, чем численное решение задач по готовым алгоритмам. Вместе с тем никто не пытается составить универсальную программу для решения всех задач численного анализа. Естественный путь, по которому идет здесь развитие вычислительной математики,— это развитие систем автоматизации программирования, включающих в себя как обширные библиотеки стандартных и типовых программ, так и средства интерпретации и трансляции с алгоритмических языков, облегчающие программирование методов решения не только массовых, но и индивидуальных проблем.

Такие области применения ЭЦМ, автоматизация проектирования или автоматизация управления в экономике также вряд ли более сложны, чем автоматизация доказательства теорем. Между тем в этих областях, где существует уже немалый опыт развития, для сколько-нибудь сложных случаев сегодня, как правило, отказываются от полностью автоматических систем в пользу автоматизированных систем, работающих в тесном контакте с человеком. При этом на долю человека приходится выбор направления поисков, постановка промежуточных целей, выбор критериев для оптимизации и окончательная оценка полученных результатов, иными словами — все, что требует интуиции и опыта. На долю же машины приходится поиск, сбор и сортировка необходимых исходных данных, фактическое вычисление критериев, окончательное оформление результатов.

Разумеется, такая ситуация не вечна. Однако сейчас, когда изучение человеческой интуиции находится в начальной стадии, а вычислительные машины плохо приспособлены для логической обработки «целостных» образов, автоматизированные системы при автоматизации сколько-нибудь сложных участков умственной деятельности могут оказаться и фактически оказываются более эффективными, чем полностью автоматические. Что касается, в частности, автоматизации доказательства теорем, то центр тяжести работы в этой области должен быть смещен от построения «универсальных» доказывающих программ в сторону создания систем автоматизации программирования и операционных систем, позволяющих, в случае необходимости, быстро программировать поиск доказательства даже одной-единственной трудной теоремы и способных, если потребуется, работать в истинном масштабе времени с математиком, доказывающим эту теорему. На таком и только на таком пути можно за разумное время добиться ощутимых практических результатов.

Первым шагом при реализации подобной программы исследований является разработка практического формального языка для записи

ма тематических предложений и их доказательств. Этот язык должен относиться к существующим формальным языкам математической логики, как, скажем, язык АЛГОЛ-60 относится к языку рекурсивных функций или нормальных алгоритмов. В этот язык должен быть включен такой общепринятый в современных практических алгоритмических языках оператор, как оператор присваивания. При этом в отличие от языков, ориентированных на вычислительные задачи, переменным должен здесь присваиваться не только числовые значения, но и значения, соответствующие различным математическим понятиям.

Операторы присваивания вида G:=группа, M:=множество хорошо согласуются с привычными для математика выражениями: «пусть G группа», «обозначим через M множество». Подобная согласованность же является главным условием практичности языка, залогом того, что он будет принят и будет использоваться всеми математиками, а не только специалистами по математической логике. Здесь опять видна аналогия с вычислительными алгоритмами. До тех пор, пока они записывались лишь в машинных языках, составление программ и пользование машинами было уделом специалистов-программистов. С развитием проблемно-ориентированных языков (типа АЛГОЛ-60) круг пользователей машин резко расширяется.

С целью увеличения степени практичности языка целесообразно расширить понятие функции, включив в него конструкции, используемые обычно для построения так называемых лексических примеров. Например, выражение «подмножество множества M» можно рассматривать как одноместную неоднозначную функцию пдм(M), аргумент которой может принимать значение «множество». В таком случае оператор присваивания N: пдм (M) есть прямой эквивалент привычного для математиков выражения «пусть N подмножество множества M». Имея в виду, что понятие функции может само явиться объектом доказательства, лучше для подобного рода функций пользоваться специальным названием, например называть их «конструкциями».

В таком случае понятия «множество», «группа» и другие можно трактовать как нульместные неоднозначные конструкции, понятие «пустое множество» — как нульместная однозначная конструкция, «подмножество M» — одноместная неоднозначная конструкция, «единица» G — одноместная однозначная конструкция. В способах образования конструкций должны существовать эквиваленты для таких выражений, как «подмножество множества M, состоящее из всех элементов α таких, что...». Само собой разумеется, что конструкции в нашем смысле пригодны и для описания объектов, неконструктивных в смысле современной конструктивной математики.

Описания конструкций должны включать в себя описание возможного типа их аргументов и типа значений конструкции. Это позволяет относительно просто строить дерево конструкций в виде суперпозиций соответствующих функций. Хорошо известно, что выбор подходящей конструкции является одним из решающих моментов, обеспечивающих успех доказательства.

Разумеется, язык должен пользоваться также понятиями предиката и высказывания. В то же время, опять-таки из соображений практичности языка, в нем целесообразно пользоваться не только обычными, но и ограниченными кванторами, представляющими собой эквиваленты выражений «для всех множеств», «существует элемент множества M такой, что...» и т. п.

После построения такого языка в проблемно-ориентированном варианте следует разработать также его машинно-ориентированный вариант и соответствующий транслятор. Желательно максимально сблизить оба варианта языка, возможно, ограничиваясь лишь пословной перекодировкой.

Вторым важнейшим шагом в осуществлении намеченной программы является построение так называемого алгоритма очевидности. В первоначальном виде алгоритм очевидности может быть ограничен простейшими функциями вывода (не обязательно естественного) в исчислении высказываний, дополненного перебором на небольшое число шагов дерева конструкций. Кроме того, в алгоритме должна быть чисто информационная часть, позволяющая осуществить поиск и вызов на специальное рабочее поле тех предложений из числа записанных предварительно в память машины, которые используются на данной попытке вывода. В дальнейшем этот алгоритм должен непрерывно дополняться и совершенствоваться.

После того как построен и запрограммирован (в машинных кодах) первый вариант алгоритма очевидности, наступает следующий этап. На этом этапе в память машины записывается какой-либо фрагмент математики, например основы теории групп с полным определением всех понятий и полными доказательствами всех предложений. Доказательство при этом считается полным, если каждый его шаг может быть проверен (по имеющейся машинной записи доказательства) машинным алгоритмом очевидности за время, не превышающее (для каждого отдельного шага) некоторый фиксированный заранее порог.

Далее разрабатывается операционная система для работы с накопленным материалом и его пополнения. Смысл этой системы в том, что с одного или нескольких пультов можно вводить в машину на построенном языке различные предложения рассматриваемой теории. Прежде всего, машина должна ответить, является ли это предложение новым или оно уже было записано ранее (возможно, ответ нужно снабдить необходимыми библиографическими данными). При положительном ответе анализируется, не является ли это предложение очевидным (в смысле машинного алгоритма очевидности) следствием уже известных машине предложений. Если нет, то система должна обеспечить возможность принятия с соответствующего пульта последовательных шагов доказательства рассматриваемого предложения, а также проверку очевидности каждого шага и уведомления математика о своем «понимании» или «непонимании» каждого очередного высказывания в доказательстве.

Уже на этом этапе система может иметь достаточно серьезное практическое значение. Во-первых, известную ценность представляет простая возможность получения справки о новизне сформулированного предложения. Поскольку информационная часть системы (список предложений и их доказательств) пополняется многими людьми, то рано или поздно каждый отдельный ее пользователь может почерпнуть из нее много полезной для себя информации. Во-вторых, строго формально и единообразно контролируется правильность вводимых доказательств и их понимание тем, кто в это время работает у пульта. Понять доказательство — это значит уметь «объяснить» его машине, работающей с достаточно бедным (но, конечно, в разумных пределах) алгоритмом очевидности. Это обстоятельство имеет существенное значение для обучения студентов и аспирантов.

Первые прикидки показывают, что уже сейчас нетрудно построить алгоритмы очевидности, сила «видения» которых будет в среднем равна

и даже превосходить силу «очевидного видения» большинства математиков. Разумеется, при этом происходит некоторая переоценка трудностей. В ряде случаев, например, при эквивалентных преобразованиях формул достаточно простые машинные алгоритмы очевидности могут превзойти способности человека. В то же время возможны и такие случаи, когда машине будут необходимы дополнительные по сравнению с человеком разъяснения. Равенство в среднем, о котором идет речь, означает, что записи доказательства, очевидных с точки зрения машины, имеют примерно такой же объем, как и их записи в форме, очевидной для специалиста.

С появлением описанной системы можно будет иметь математические журналы, в которых в число формальных правил для публикаций будет входить возможность машинной проверки доказательств. Публикации же в обычных журналах могут и будут разбираться специалистами, работающими с системой, таким образом, чтобы они стали понятными (и потому правильными) не только для них самих (что всегда субъективно и ненадежно), но и для машины.

Появляется возможность перестройки изложения всей математики на действительно современной основе, так как фундаментальный труд Бурбаки при всех его неоспоримых достоинствах все же не до конца современен, поскольку в нем не используются достижения современной машинной математики для действительно полного и исчерпывающего изложения основ математики с единым стандартом очевидности и с полной уверенностью в отсутствии ошибок в технике доказательств.

Дальнейшее развитие системы связано с развитием и усовершенствованием алгоритма очевидности. Он должен дополняться все время пополняющейся библиотекой стандартных процедур вывода. Прежде всего это касается таких вопросов, которые хотя и не очевидны (в смысле прямого и непосредственного видения) математику, но детали вывода которых приводить обычно не принято (например, взятие интеграла от достаточно сложной рациональной дроби).

Далее, должны быть разработаны специальные языковые средства с системами трансляции или интерпретации, которые давали бы возможность программировать поиск доказательства отдельных трудных теорем. Эти средства должны помогать усовершенствованному алгоритму очевидности прежде всего в построении дерева конструкции. Для этого они должны иметь возможность использовать для ограничения перебора указания типа: «применить прием (или конструкцию), аналогичный приему в доказательстве теоремы такой-то», «упорядочить множество и использовать индукцию», «рассмотреть отдельно случаи...» и т. п.

Соответствующим образом должен быть также разработан язык, на котором машина сообщает математику о тех трудностях, которые мешают найти доказательство. Само собой разумеется, что машина должна выдавать в заключение текст доказательства, пригодный для публикации.

Существенно также то, что информационная часть системы (совокупность известных машине результатов и их доказательств) все время растет, охватывая все новые и новые области математики. Причем дело не сводится лишь к простому количественному росту объема «машинных знаний». С каждым новым запомненным результатом система перестраивает структуру своей информационной части. Действительно, наличие нового результата может сделать очевидными (с точки зрения машины) некоторые теоремы или некоторые разделы доказательств других теорем, которые раньше требовали более подробной расшифровки. То же происходит при каждом усовершенствовании алгоритма очевидности, при улучшении машины или

нахождении более эффективных приемов машинной обработки информации (поиска, сортировки, редактирования и т. п.).

Появляется возможность объективной количественной оценки «обобщающей силы» каждого нового математического результата. В качестве такой оценки может использоваться сокращение объема информационной части системы за счет исключения некоторых доказательств и упрощения других. Любопытно, что при этом появляется возможность «взвешивать на одних весах» упрощения, вносимые в обоснование математических теорий как самыми абстрактными новыми теоремами, так и усовершенствованиями программ системы (прежде всего алгоритма очевидности) и даже усовершенствованием самой вычислительной техники.

Разумеется, следует иметь в виду, что обобщающая сила нового научного результата является лишь одной из компонент в оценке его важности. Не менее, а, пожалуй, даже более существенно то, что новые результаты могут открывать новые области исследований и новые, не изучавшиеся ранее, свойства, важные с точки зрения расширения сферы приложений математики. Однако в современный период, когда роль обобщений в развитии математики достаточно велика, возможность объективной оценки их важности представляет несомненный интерес.

В заключение несколько слов о более далеких перспективах развития намеченной программы автоматизации доказательств. Непрерывное совершенствование алгоритма очевидности рано или поздно приведет к тому, что все теоремы, которые нам известны сегодня, станут очевидными с точки зрения машины. В этот период роль математика будет состоять преимущественно в определении новых понятий и в формулировке принципиально новых предложений, а искусство доказать новую, не очевидную с точки зрения машины теорему будет состоять прежде всего в умении сформулировать ряд промежуточных теорем и лемм, каждая из которых очевидна для машины. Что же касается результатов чисто обобщающего характера, то, имея объективный критерий для их отбора, задачу поиска их формулировок и доказательств можно поручить самой системе.

РАСПОЗНАВАНИЕ ОБРАЗОВ В БИОНИКЕ

(Бионика, М.: Наука.— 1965)

Задача распознавания образов принадлежит к числу важнейших и вместе с тем наиболее трудных задач бионики.

Важность этой задачи обусловливается чрезвычайно большим многообразием приложений. Это не только распознавание зрительных образов или звуков речи. Сегодня проблема распознавания образов включает в себя такие вопросы, как, например, распознавание производственных ситуаций, синтаксически правильных построений в естественных и искусственных языках, распознавание смысла фраз и т. п.

Однако проблема распознавания зрительных образов до настоящего времени продолжает занимать ведущую роль в общей проблеме распознавания образов как по своему практическому значению, так и по глубине разработки вопроса. Практический аспект проблемы касается прежде всего задачи автоматизации ввода в электронные цифровые машины цифровой и буквенной информации, чертежей, рисунков, фотографий с помощью так называемых читающих автоматов. Разумеется, далеко не все работы по читающим автоматам можно отнести к бионике. В частности, интересные работы В. А. Ковалевского [8] не являются в полной мере бионическими, хотя и приводят к хорошим практическим результатам (надежность распознавания по сравнению с обычным человеческим восприятием повышается на порядок).

Чисто бионический аспект проблемы прежде всего лежит в области распознавания нестандартных изображений на основе тех или иных обучающих процедур. Первой существенной трудностью, появляющейся при таком подходе, является огромное количество возможных образов. Хорошо известно, что для сетчатки, состоящей из N двухпозиционных элементов, общее количество различных образов равно 2^{2^N}. Поскольку при постановке вопроса в приемлемой для практики форме величина N должна измеряться по крайней мере сотнями, то количество всех возможных образов будет чудовищно велико.

Поэтому неудивительно, что были предприняты попытки ввести те или иные ограничения, которые позволяли бы резко уменьшить число подлежащих рассмотрению образов. Так, Браверман [2] предложил ограничиться рассмотрением лишь так называемых компактных образов. К сожалению, в случае сетчатки, составленной из двух позиционных элементов, компактных образов практически (за исключением случая с образами, состоящими из всех или почти всех изображений) не существует. Более ударной в этом смысле является гипотеза N-экстраполируемости образов, предложенная автором [3].

Ясно, однако, что сужение понятия образа в направлении максимального приближения его к образам, легко распознаваемым человеком, требует более тонкого анализа свойств нейронных сетей в соответствующих

распознающих центрах головного мозга. Подобное семейство «человечности» образа зависит, разумеется, от вида восприятия: так, легко распознаваемые человеком звуковые образы могут оказаться трудно распознаваемыми при переводе их (тем или иным фиксированным способом) в зрительную форму, и наоборот. Не исключено, что систематическое изучение и использование подобной избирательности органов чувств позволит в некоторых случаях упростить решение задачи обычного (неавтоматического) распознавания. Возможно, например, что представление электрокардиограмм или энцефалограмм в звуковой, а не в визуальной форме позволит быстрее и точнее распознавать скрытые пороки работы сердца или мозга.

Решение же проблемы автоматического распознавания образов в бионическом плане тесно связано с задачей моделирования сетей нейронов.

Не следует думать, разумеется, что при таком моделировании нужно слепо копировать природу абсолютно во всех ее проявлениях. Разумный уровень точности моделирования прежде всего определяется технологическими факторами. В частности, точность моделирования нервных сетей может и должна повышаться соразмерно с прогрессом технологии изготовления радиоэлектронных устройств. Поэтому, не рассматривая пока более тонкие свойства обычных «живых» нейронов, можно ограничиться изучением сетей нейронов в том смысле, как они понимаются в современной теории автоматов. При этом каждый нейрон представляется в виде порогового элемента с непрерывными либо с двоичными входными и выходными сигналами.

Возникающие подобным образом идеализированные нервные сети можно разделить на полностью дискретные, так называемые логические сети и на сети, часть параметров которых может пробегать то или иное непрерывное множество значений (чаще всего такое множество составляют все вещественные числа, заключенные между 0 и 1). Отметим, что в задаче распознавания образов полностью дискретным цепям до сих пор уделялось недостаточное внимание. Между тем этот вопрос заслуживает более тщательного изучения.

Дело в том, что булевы функции «нейронного» типа, реализуемые пороговыми элементами, обладают рядом свойств, интересных с точки зрения задач распознавания образов. Действительно, любая система булевых функций, состоящая из n функций $f_1, f_2, \ldots, f_n$ от n переменных $x_1, x_2, \ldots, x_n$, задает некоторое преобразование двоичной n-элементной сетчатки в себя. При каких условиях такое преобразование будет взаимно-однозначным? Легко понять, что одним из необходимых условий является равенство друг другу числа единиц и числа нулей в таблице значений любой из функций $f_1, f_2, \ldots, f_n$. Этому условию удовлетворяет всякая пороговая булева функция с $2m - 1$ входными каналами, если число q запрещающих входов и порог p удовлетворяют соотношению $q + p = m$ ($m = 1, 2, 3, \ldots$).

Далее, из пороговых элементов можно строить правильные (многослойные) цепи, осуществляющие перевод ряда характерных для «человеческих» зрительных образов признаков из сложно кодируемой (на исходной сетчатке) формы в просто кодируемую. Возникает естественное предположение, что «человечность» образов связана с простотой кодирования характерных для них признаков в верхних слоях соответствующих распознающих нейронных сетей. Условие же взаимной однозначности преобразования в каждом слое (а, следовательно, и во всей сети) сохраняет

возможность распознавания (хотя и более трудным путем) сложно кодируемых признаков непривычных для человека образов.

В связи с естественным требованием простоты исходных гипотез о закономерностях строения распознающих сетей особый интерес представляет задача изучения случайно организованных многослойных цепей нейронов. При этом изучение регулярных (правильно организованных) сетей делает естественным предположение, что достаточно хорошие результаты должны получиться в сетях, где вероятности подсоединения входов каждого нейрона $(i+1)$-го слоя к выходам нейронов i-го слоя распределены не по равномерному, а по нормальному закону (слои предполагаются лежащими один над другим в виде прямоугольных площадок достаточно больших размеров, чтобы было возможно пренебречь краевым эффектом).

Еще одна особенность нейронных сетей, заслуживающая внимания с точки зрения распознавания образов,— это возможность простого осуществления однородного глобального механизма регулирования среднего уровня активности нейронов, но так, чтобы в каждом случае активной была бы примерно половина всех нейронов сети. В нейронных сетях достигнуть этого можно за счет одновременного изменения в ту или иную сторону порогов возбуждения всех нейронов. Точное выполнение условия половинной активности потребует, разумеется, перехода от дискретных к непрерывным сетям.

Интересные результаты можно получить также при использовании для целей распознавания образов моделей, основанных на идеях машины условной вероятности [1] и машины условных рефлексов [5]. В задачах распознавания синтаксических и семантических (языковых) образов особую роль приобретает понятие регулярного события и его связи с понятием конечного автомата.

Что же касается зрительных образов, то бионический аспект задачи их распознавания наиболее ярко проявляется в развитии работ по теории персептрона, вступившей сейчас в новый этап своего развития. Как известно после первых работ Ф. Розенблатта, породивших большие надежды, наступила пора разочарований. Схема трехслойного персептрона, предложенная Розенблаттом, оказалась весьма ограниченной в своих возможностях, а анонсированные им результаты в большей части — ошибочными.

В работах Джозефа [9], В. М. Глушкова [4, 5], и Розенблатта [1] были получены следующие результаты, объяснившие с достаточной точностью возможности трехслойных персептронов.

Во-первых, в значительной мере была развенчана гипотеза о существовании у персептронов спонтанной независимой организуемости, так называемого режима самообучения.

Во-вторых, показана недостаточность первоначально предложенных Розенблаттом механизмов поощрения в режиме обучения и необходимость перехода к системам с коррекцией ошибки.

В-третьих, установлено, что для возможности осуществления произвольной классификации в трехслойных персептронах требуется огромное число A-элементов (не менее 2^n при двоичных рецепторах).

В-четвертых, оказалось, что хотя при переходе на режим обучения с коррекцией ошибок всякая допустимая бинарная классификация может быть достигнута при обучении случайными последовательностями достаточно общего вида. Но при этом требуется, как правило, неоправданно большое число экспериментов.

В-пятых, установлено полное отсутствие предпочтительности осуществляемых персептронами классификаций к естественным критериям подобия (возникающим на основе применения к изображениям преобразований геометрического подобия, переносов и поворотов).

В-шестых, выявилось полное отсутствие характерной для человека возможности обучения по частям, т. е. первоначального обучения простым свойствам, а затем на их базе — сложным. Более того, в случае нескольких R-элементов (что совершенно необходимо для обучения по частям) при использовании систем с коррекцией ошибки отсутствует, вообще говоря, свойство достижимости всякой априори возможной классификации.

Работы, о которых говорилось выше, не только определили действительные границы возможностей трехслойных персептронов, но и наметили пути дальнейших исследований и усовершенствований. В результате исследования по теории персептронов вступили в совершенно новую фазу, когда изучаемые модели начинают воспроизводить многие глубокие и тонкие особенности работы распознающих первых центров человеческого мозга. Весьма интересные результаты в этом направлении получены Розенблаттом и некоторыми другими авторами. Эти результаты, обобщенные в монографии Розенблатта [10], не привели еще к созданию связной теории и ограничиваются пока экспериментальным материалом и отдельными теоретическими заключениями.

Главные направления исследований по персептронам, ведущихся в настоящее время, следующие.

1. Переход к изучению многослойных систем, а также систем с пересекающимися и возвратными связями. Одна из первых работ такого рода была выполнена автором совместно с В. А. Ковалевским и В. И. Рыбаком [7]. Многочисленные исследования в указанном направлении выполнены Розенблаттом.

2. Введение механизмов управления порогом A-элементов [10].

3. Переход от построения обучающих последовательностей по методу независимых испытаний к построению их как марковской цепи. При этом возникает возможность самообучения как процесса, переводящего временную близость изображений в их пространственную близость [4, 10].

4. Переход от распознавания отдельных изображений к распознаванию последовательностей изображений, что позволило включить в круг «персептронной» тематики не только зрительные, но и звуковые анализаторы. Делаются попытки объединить зрительный и звуковой персептроны в единую распознающую систему [10].

Принципиально новым здесь является переход к распознаванию пар образов, чем по сути стирается различие между режимами обучения и самообучения.

Любопытно, что в персептронах с перекрестными связями или с двумя слоями A-элементов возникает возможность обучения группе преобразований (на практике чаще всего локальной группе), сохраняющих принадлежность образу. Закономерность проторения связей для одного изображения как бы проторяет пути для быстрого усвоения таких преобразований для других изображений. Намечается достаточно простое объяснение многих характерных для человека особенностей преобразований, не меняющих принадлежность изображения или иному образу.

Персептроны с перекрестными связями оказывают гораздо более экономными (по сравнению с обычными трехслойными персептронами) с точки зрения расходования A-элементов. Камерон аноисировал результат о том, что в подобном персептроне все возможные классификации могут

быть реализованы при общем числе A-элементов, не превосходящих числа рецепторов (двоичных) сетчатки. В персептронах с перекрестными связями достаточно естественно реализуются процессы распознавания последовательностей.

При переходе к персептронам с возвратными связями происходит дальнейшее сближение с распознавательными возможностями, присущими человеку. Проявляется, в частности, свойство концентрации внимания и возможность обучения по частям (Rosenblatt, 1962).

Как известно, обучение распознаванию понятий и смысла фраз в естественных языках [3—6] до настоящего времени строилось методами, которые по своему оформлению были довольно далеки от «персептронной» тематики. Сейчас в результате расширения понятия персептрона (прежде всего за счет введения возвратных связей) создалась возможность объединить эти работы с работами по распознаванию зрительных образов. Весьма заманчивой является идея объединения в единой модели на основе использования возвратных связей от R-элементов трех персептронов, распознающих зрительные, звуковые и логические (смысловые) образы. На этом пути можно будет, по-видимому, построить первую (хотя пока еще весьма примитивную) глобальную модель мозга.

Возможно, что на данной основе удастся подойти к вопросу о конструировании наиболее естественного «внутреннего» языка для моделирования процессов мышления. В таком языке система кодирования понятий была бы, несомненно, гораздо более близкой к системе кодирования непосредственных зрительных и звуковых восприятий. А это, в свою очередь, упростило бы задачу дешифровки в зрительных и слуховых распознающих центрах.

СПИСОК ЛИТЕРАТУРЫ

1. *Аттли А. М.* Машины условной вероятности и условные рефлексы // Автоматы. М.— 1956.
2. *Браверман Э. М.* Опыты по обучению машины распознаванию зрительных образов // Автоматика и телемеханика.— 1962.— № 3.
3. *Глушков В. М.* Введение в теорию самосовершенствующихся систем. Киев.— 1962.
4. *Глушков В. М.* Теория обучения одного класса дискретных персептронов // Журн. вычислительной математики и математической физики.— 1962.— 2, № 2.
5. *Глушков В. М.* К вопросу о самообучении в персептроне // Журн. вычислительной математики и математической физики.— 1962.— 2, № 6.
6. *Глушков В. М., Грищенко Н. М., Стогний А. А.* Алгоритм распознавания осмысленных предложений // Принципы построения самообучающихся систем. Киев.— 1962.
7. *Глушков В. М., Ковалевский В. А., Рыбак В. И.* Алгоритмы обучения машины распознаванию простейших геометрических фигур // Принципы построения самообучающихся систем. Киев.— 1962.
8. *Ковалевский В. А.* Корреляционный метод распознавания изображений // Журн. вычислительной математики и математической физики.— 1962.— 2, № 4.
9. *Jozef R. G.* On predicting perceptron performance // IRE. Intern. Conv. Rec.— 1960.— N 2.
10. *Rosenblatt F.* Principles of neurodynamics. Washington.— 1962.

ПРОБЛЕМА АВТОМАТИЗАЦИИ ДЕДУКТИВНЫХ ПОСТРОЕНИЙ

(Управление. Информация. Интеллект.— М. : Мысль, 1976)

Автоматизация дедуктивно-эвристической деятельности — важная область исследований по искусственному интеллекту, представляющая значительный интерес для совершенствования управления познавательными и технологическими процессами и повышения их эффективности. Однако, как мы убедились, настоящего практического значения в этом направлении данные работы еще не получили.

В чем «слабость» той ситуации, которая имеет место ныне в автоматизации дедуктивных построений? Это прежде всего естественное на первых порах стремление к некоей универсальности инструмента автоматизации. Чтобы автоматизировать, например, построение доказательств сложных теорем в математике, разрабатывают некую «единую» доказывающую программу. Такого рода программа основывается на базе современной логики, реализуя ту или иную форму поиска логического вывода. На данном пути уже достигнуты обнадеживающие результаты, однако лишь в случаях, когда эти доказывающие программы применяются в рамках простейших математических теорий и прежде всего к самой математической логике.

Вопрос здесь в том, что автоматизация дедуктивных построений в целом представляет собой **очень сложную задачу** — более сложную алгоритмически, чем, например, задачи численного решения уравнений математической физики. И никто не пытается построить универсальную **вычислительную** программу, с помощью которой можно было бы решать любые задачи вычислительной математики. Тем более было бы странно, если б удалось построить одну-единственную **доказывающую** программу, которая обеспечила бы автоматизацию заведомо более сложной работы, чем вычисления,— автоматизацию **дедуктивных построений**, в том числе **доказательство сложных теорем.** Поэтому представляется, что для организации действительно широкого и последовательного наступления на проблему необходим существенно иной подход к проблеме. Мы познакомимся с элементами новой методологии автоматизации дедукции, разработанной в Институте кибернетики АН УССР. Сразу же отметим, что данный вопрос тесно связан не только с дальнейшим развитием работ в области искусственного интеллекта, но и с **повышением возможностей электронно-вычислительных машин.**

1. Новый подход к автоматизации дедуктивных построений. «Алгоритм очевидности». Прежде всего необходимо создать адекватный язык для представления понятий и процессов доказательств в дедуктивных науках (в математике, теоретической физике и др.), предназначенный не для программирования, а для описания самого объекта автоматизации. Имеющийся язык математической логики (при ее приложениях) в принципе пригоден и для такого рода функций, но математик или естествоиспытатель практически никогда им не пользуется — он неудобен для применения.

Поэтому задача состоит в том (и исследования в этой области уже проводятся), чтобы построить практически-ориентированный язык математической логики. Этот язык должен относиться к «классическому» языку математической логики примерно так, как алгоритмический язык типа АЛГОЛ относится к языку, предположим, машин Тьюринга, или алгоритмов Поста, или нормальных алгоритмов Маркова.

И АЛГОЛ, и «финитные комбинаторные процессы» Поста, и нормальные алгоритмы являются универсальными алгоритмическими языками, и в принципе на каждом из них можно выразить любой алгоритм. Но как мы уже отмечали, язык Поста, как и язык Маркова и Тьюринга, строится из мельчайших составных «кирпичиков», и сложить из них здание реального алгоритма чрезвычайно трудно, в то время как АЛГОЛ пользуется гораздо более крупными, привычными для математика блоками. В этом и состоит существенное различие между указанными языками: АЛГОЛ — это практически-ориентированный алгоритмический язык, а языки Маркова и Поста — «теоретические». Последние строились для решения проблем основ математики, и стремление к простоте языка, к минимизации числа его элементов, к уменьшению размеров самых элементов было естественным. Это облегчало изучение возможностей самого языка и его использование при формализации дедуктивных построений в целях исследования самых фундаментальных основ математики как «доказывающей» науки.

Чем же должен отличаться практически-ориентированный логический язык от языка обычной математической логики? Прежде всего тем, что с его помощью можно будет оперировать более крупными «блоками» доказательств.

Поясним идею «крупных блоков» на простом примере. Далее приведена блок-схема алгоритма поиска доказательств теорем в исчислении высказываний. Основные принципы работы алгоритма следующие. Целью (Ц) является формула исчисления высказываний, которую на данном шаге работы алгоритма необходимо доказать.

Посылка (П) — формула высказываний, в предположении истинности которой доказывается соответствующая цель. Главными шагами алгоритма являются расщепление цели и построение вспомогательной цели. При расщеплении из цели выделяются некоторые, другие более простые по структуре цели, из которых непосредственно следует доказываемая цель и, возможно, посылка. Цель считается доказанной, если существует посылка, с ней совпадающая. В этом случае вспомогательная цель будет пустой.

Работу данного алгоритма иллюстрирует следующий пример (рисунок).

Теорема Ц1 ((A C) ((B C) ((AB) C))). Для доказательства этого утверждения предположим:

П1 (A C)

и докажем:

Ц2 ((B C) ((AB) C)).

Для доказательства этого утверждения предположим:

П2 (B C)

и докажем:

Ц3 ((A B) C).

Для доказательства этого утверждения предположим:

П3 C

и докажем:

Ц4 (A B)).

Внося отрицание, получаем:

Ц5 (A B).

Данное утверждение будет доказано, если мы докажем

Ц6 А и Ц7 В.

Докажем последнюю цель

Данное утверждение будет доказано, если мы докажем получаемую из посылки П2 вспомогательную цель:

Ц8 С.

Данное утверждение непосредственно вытекает из посылки П3. Тем самым данная ветвь доказана. Переходим к доказательству утверждения

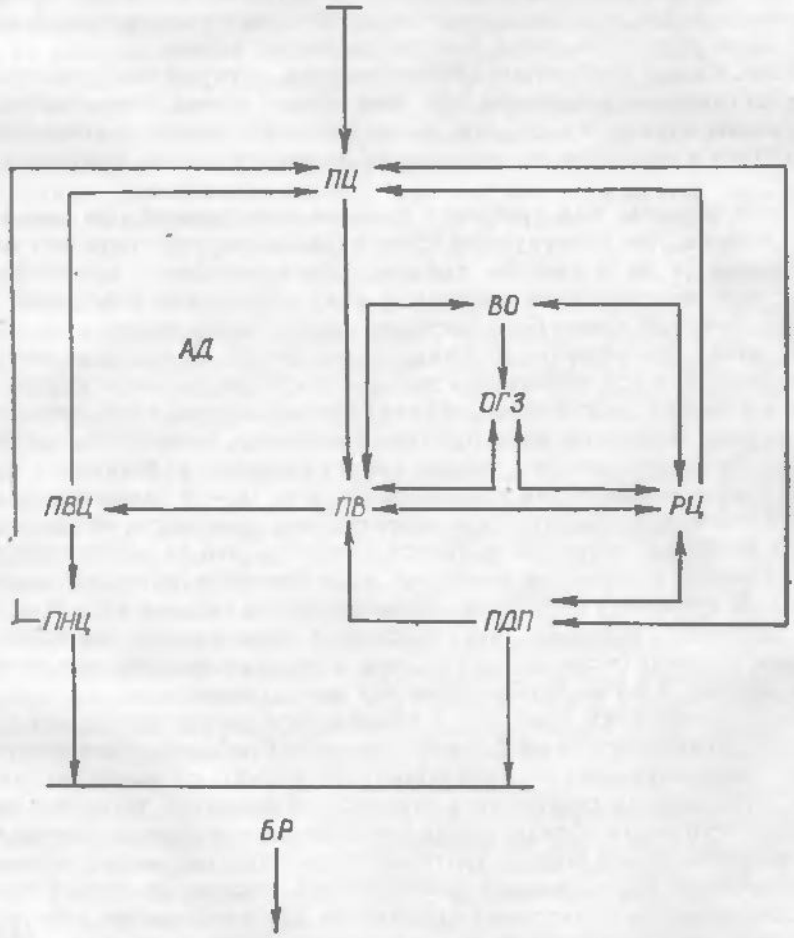
Ц9 А.

Данное утверждение будет доказано, если мы докажем получаемую из посылки П1 вспомогательную цель:

Ц10 С.

Данное утверждение непосредственно вытекает из посылки П3. Тем самым теорема полностью доказана.

Как мы видим, задание схемы этого алгоритма сделано на языке крупных блоков. Для реализации блоков существуют стандартные подпрограммы. Это означает, что блоки понимаются машиной, и, следователь-



Блок-схема алгоритма поиска доказательств теорем в исчислении высказываний:

АД — блок диспетчера алгоритма; ПВ — поиск вхождения цели в посылку; ПВЦ — построение вспомогательной цели; РЦ — расщепление цели; ВО — внесение отрицания в формулу; ОГЗ — определение главного знака формулы; ПЦ — проверка цели; БР — блок редакции доказательства; ПНЦ, ПДП — режим работы блока АД при поиске соответственно недоказанной цели и при поиске другого пути.

но, программист может их использовать при программировании задач
Ясно, что подобное программирование резко отличается от программиро
вания, скажем, на языке машины Поста.

Кроме обращения к «крупным блокам» в практически-ориентирован
ном логическом языке следует предусмотреть средства определения новы
понятий. Это возможно, так как язык математической логики, к котором
добавлены требуемые постоянные предикаты и функции, позволяе
выразить соответствующей формулой то или иное понятие.

Допустим, необходимо определить какой-то класс функций, задава
их специальные свойства. Понятие об этом классе функций можно разло
жить на элементы и представить определенным выражением логико-матема
тического языка. Но математик, рассуждая и проводя соответствующи
записки, как правило не пользуется таким приемом — не оперируе
подобным выражением. Он прибегает к сокращенному обозначению
общему «образцу» данного класса функций. К строгому определению о
будет обращаться лишь по мере необходимости, «вытаскивая» из него т
или иные свойства, нужные для его эвристических и дедуктивных постро
ений. Такого рода «образцы» и должны вводиться в язык.

Далее, в язык необходимо ввести средства, которые позволяют поро
дать, конструировать примеры тех или иных общих соотношений. Э
тоже важно, однако чуждо для «классического» языка математическо
логики (хотя в принципе на нем можно конструировать требуемые пр
меры).

Таким образом, нам требуется понятие конструкций при доказател
ствах. Отметим, что конструкция здесь понимается в обычном для матем
тики смысле (а не в смысле, скажем, конструктивной математики),
именно: под конструкцией имеется в виду объект или некоторый кла
объектов, который может быть построен либо конструктивно в подлинно
смысле, либо с помощью тех или иных определений. Вся суть математиче
ких построений и вся «изюминка» доказательств (не только в математиче
ких, но и в любых дедуктивных науках) состоит как раз в том, что исслед
ватель строит некоторую конструкцию. Например, математик рассуждае
«Возьмем функцию, которая удовлетворяет таким-то условиям на конца
таким-то ограничениям для производной» и т. п. — и начинает из клас
функций с помощью различных методов строить подклассы, выделяя те и
иные их свойства; затем он пытается доказать, что та конструкция, т
класс функций, который он построил, удовлетворяет условиям некотор
теоремы. И трудность построения доказательства состоит не в том, чтоб
путем дедукции доказать, что найденная конструкция удовлетворя
условиям теоремы (а это именно то, что и делала классическая матема
ческая логика), а в том, чтобы найти эту конструкцию.

Когда математику приходит в голову идея конструкции, это значи
что он интуитивно чувствует близость решения проблемы; идея «запуска
процесс доказательства — начинается дедукция: с помощью аксио
теорем, предыдущих результатов строится обоснование того, что предл
женная конструкция обладает теми свойствами, которые заложены в ид
самой теоремы. Такой подход противоположен подходу «обычной» матем
тико-логической формализации доказательств средствами логики предик
тов: использование исчисления предикатов для нахождения конструкц
совершенно не естественный для мышления математика путь. Поэто
«чистая» математическая логика не дает возможности по-настоящ
автоматизировать процессы доказательств теорем (кроме достаточ
узкого круга теорий, включая теории за пределами самой математическ

логики), и приходится создавать различного рода эвристические программы. Таким образом, реалистический подход к автоматизации дедуктивных построений означает прежде всего выработку нового практически-ориентированного математико-логического языка (и начальный вариант такого языка в настоящее время уже построен).

Вторым компонентом предлагаемой программы работ — компонентом, связанным уже с техникой самих доказательств, является так называемый **алгоритм очевидности**. Суть проблемы в следующем. Не ставится задача построения такого алгоритма, следуя которому машина доказывала бы все теоремы (некоторой теории). Машина должна доказывать очевидные вещи: **не угадывать конструкции, а по готовой конструкции проверять**, удовлетворяет ли она тем или иным свойствам, применяя для этого логический аппарат, который использует «крупные блоки» дедуктивных выводов, заложенные в ранее доказанных теоремах. Характер этого алгоритма очевидности в принципе ясен.

И третьим компонентом рассматриваемого подхода (вплотную примыкающим к предыдущему) является **информационная база**. Когда есть язык, на котором можно записывать определения, аксиомы, алгоритмы, формулировки теорем, их доказательства и т. д., то возникает естественное желание записать на этом языке тот или иной раздел математики. В работах коллектива математиков Института кибернетики АН УССР в качестве такого раздела выбрана, например, алгебра.

Активно разворачивающиеся работы по логической перестройке математики охватывают преимущественно ее «нижние этажи»: арифметику (теорию чисел), ряд разделов алгебры, основы математического анализа. Трудности «движения вверх» связаны с тем, что математик далеко не всегда в состоянии фактически довести доказательство теоремы до разложения его на те элементарные «кирпичики», которыми располагает обычная математическая логика, ведь «длина» доказательств может при этом увеличиться в сотни раз, и читать такие доказательства человеку будет, вероятно, не под силу. Труд, при котором доказательства теорем большого «куска» математики представлялись бы в языке математической логики, теперь фактически не затрачивается. Это приводит к тому, что математики, в том числе автор многотомного трактата по современной математике Бурбаки, злоупотребляют термином «очевидно».

Слово «очевидно» как математический термин имеет известную эмоциональную окраску: то, что очевидно одному ученому, не очевидно другому. Это связано с серьезной опасностью. «Здание» математики становится все более и более длинным, специализация все более и более узкой. Очень часто в базисных теоремах встречаются цепи доказательств настолько длинные, требующие, как правило, такой специализированной техники проверки, что выполнить эту проверку может очень небольшое количество исследователей. Например, работу, которая была проделана по одной из проблем Гильберта, поняли не более 20 человек в мире (теорема доказывается на нескольких десятках страниц, и более короткого доказательства ее не найдено). В работе Томпсона по решению малой проблемы Бернсайда доказательство одной лишь теоремы занимает 250 страниц. Ясно, что при этом вполне может где-то «проскочить» ошибка, и опасность этого кроется именно в понятии «очевидно», ведь исследователи, которые будут пользоваться теоремой, могут прийти к ложным выводам.

Сказанное свидетельствует о важности такого подхода к решению проблемы автоматизации дедуктивных построений, при котором математическое «здание» покоилось бы на вполне понятном, объективированном,

едином, универсальном понятии очевидности. Построение некоего алгоритма очевидности и операционной системы организации вычислительного процесса с информационной базой на той или иной конкретной вычислительной машине будет означать введение формального практического понятия очевидности (теоретическое понятие «очевидности» фактически присутствует в математической логике, но реально им пользоваться невозможно). Под «формальное практическое понятие очевидности» подпадает всякий вывод, который может быть получен в результате определенного времени (предположим, не более минуты) работы алгоритма очевидности на некоторой машине (предположим, БЭСМ-6 в определенной комплектации). Такие выводы и образуют формальное определение очевидности.

В отношении самого «алгоритма очевидности», конечно, можно спорить, можно его улучшать и т. д. После того как какой-то алгоритм очевидности зафиксирован, возникает возможность построить и записать «фундамент математики» (условно будем считать — по Н. Бурбаки), в котором все базисные понятия определены точно, а все базисные теоремы строго доказаны. В этих доказательствах будет применено понятие очевидности, однако оно должно использоваться не в таком «измельченном» виде, как при выводах в обычной математической логике. Вместе с тем это понятие будет совершенно объективным: все доказательства с ссылкой на «очевидность» можно будет проверить на машинах. А далее возникает возможность в соответствии с этим алгоритмом очевидности перестроить остальные ветви математики.

2. Пути реализации и ближайшие возможности. Хотя мы находимся еще довольно далеко от реализации предлагаемой системы автоматизации дедуктивных построений, имеет смысл уже сейчас осознать, что может нам дать практика ее создания. Допустим, в нашем распоряжении есть некоторая вычислительная система с разделением времени, обладающая соответствующими пультами (на которых могут работать исследователи), операционной системой, организующей вычислительный процесс, и соответствующей информационной базой. Предположим, например, что в эту базу заложена вся современная алгебра — все теоремы из монографий и статей ведущих ученых, причем это сделано в таком виде, что каждый шаг доказательства, переход от одного шага к другому автоматически проверяются с помощью машинного алгоритма очевидности (который, естественно, туда введен). Кроме того, в машину заложена информационно-логическая система поиска, дающая ответ на вопрос о том, известна ли данная теорема или нет, и если не известна, то не получается ли она как следствие из уже известных теорем в результате работы алгоритма очевидности.

Такая система уже на первом этапе будет обладать рядом интересных и практически важных свойств. Так, она сможет контролировать обучение математиков (например, студентов или аспирантов), которым дано задание изучить какие-то новые статьи. Обучаемые, конечно, должны понять содержащиеся в статьях доказательства; понять автора — это означает расшифровать соответствующий математический текст и привести содержащееся в нем рассуждение в соответствие с принятым понятием очевидности.

При использовании «алгоритма очевидности» создается возможность осуществлять объективный контроль за пониманием. Для этого изучающий может поступать, например, следующим образом: ввести проверяемую теорему вместе с ее доказательством в машину в таком «препарированном» виде, чтобы последняя на каждом шаге работы алгоритма очевидности отвечала — это «понятно», а это «не понятно». При отрицательном ответе

человек должен разъяснить машине соответствующий дедуктивный переход, и, пока это им не сделано, доказательство будет не понятным. Это уже объективная проверка усвоения материала, причем проводимая автоматически. Описанная процедура полезна еще и в том отношении, что каждая «расшифровка» новой теоремы, получаемой из вновь вышедших работ, будет обогащать информационную базу системы: в систему будет вкладываться новый результат, которым уже на уровне машинной очевидности сможет пользоваться любой исследователь.

Еще более ценной окажется возможность следующего применения системы. Обладая соответствующим практически-ориентированным логическим языком и алгоритмом очевидности определенной силы, с помощью этой системы мы сможем организовать автоматическую проверку правильности самих новых результатов (а не только правильность их усвоения «расшифровщиками»). Для этого достаточно ввести некоторый практический язык, очень близкий к тому, который обычно используется в математических текстах, но включающий, естественно некоторые ограничения на выбор формулировок. Тогда, например, можно будет в математический журнал принимать статьи только на этом языке и автоматически (на машине) проверять правильность выкладок их авторов.

При соединении такой вычислительной системы с мощной информационной базой появляется возможность по-новому строить справочно-информационные системы. В обычном исполнении такие системы на сегодня лишь сообщают, известен им некоторый факт или нет. Но если система будет дополнена логическим механизмом, включающим алгоритм очевидности, то это уже позволит давать существенно содержательные ответы, например сообщать, что хотя запрашиваемый факт машине не известен, он является очевидным (в смысле данного машинного алгоритма очевидности) следствием таких-то фактов.

И, наконец, указанная система (точнее, ее разработка) создаст как бы «полюс концентрации» усилий большого количества исследователей (в первую очередь математиков и специалистов в области программирования и машинной техники), что неизбежно приведет к расширению возможностей самой системы. Во-первых, исследователи, работая на пультах системы, одновременно будут обогащать ее информационную базу. Таким образом, система будет «растущей» — в ее памяти будет накапливаться все большее количество фактов. Во-вторых, алгоритм очевидности, будучи опубликованным, станет достоянием большого количества людей, работающих в разных местах и способствующих обогащению этого алгоритма на основе решения конкретных задач.

Отметим, что для реализации этой программы при наличии хорошей организации операционной системы следует предусмотреть развитие и специальных средств автоматизации программирования, ориентированных на определенные классы задач (т. е. соответствующие языки и системы трансляции и интерпретации). Более того, на этом пути естественно возникнут новые требования к конструкциям машин, касающиеся таких их свойств, как удобство использования вычислительной системы, а также эффективность процессов программирования и решения на ней задач дедуктивных построений. Это значит, что в процесс развития математики и других дедуктивных наук включаются силы, которые ранее в развитии «теоретического знания» не участвовали (программисты, создатели операционных систем и даже конструкторы ЭВМ), и от всех них будет в какой-то мере зависеть дальнейшее развитие алгоритма очевидности в каждой отдельной дедуктивной дисциплине.

Следующий шаг развития «алгоритма очевидности» будет заключаться в том, что алгоритм сам начнет строить новые дедуктивные конструкции — вначале методом простого перебора, а затем используя те или иные ограничения в соответствии с направляющими указаниями программиста-исследователя, работающего за пультом вычислительной системы. На этом этапе потребуется еще один язык — язык общения машины с исследователями. С помощью этого языка будет производиться человеко-машинный поиск программирования доказательств конкретных теорем. В принципе, поскольку дедуктивные конструкции фактически представляют собой функции, аргументами и значениями которых служат математические понятия, их можно строить методом обычной суперпозиции, следя при этом, чтобы представляемые аргументы функций были такого же типа, как и функции; таким образом можно будет организовать процесс построения деревьев дедуктивных конструкций.

Далее, поскольку случайный перебор (хотя и чисто автоматический даже при очень мощных машинах) «захлебывается» уже на довольно коротких «дистанциях», необходима определенная организация автоматического перебора, т. е. сужение класса применяемых конструкций. Для этого исследователь обычно использует свои содержательно-интуитивные представления, которые он приобретает в процессе опыта. Поэтому необходимо, чтобы в языке диалога с машиной была заложена возможность сообщения машине этих интуитивных результатов, например требование выбора из некоторого класса конструкций, применявшихся при доказательствах таких-то теорем. И когда в алгоритм очевидности будет введена способность автоматического построения новых дедуктивных конструкций, он станет чрезвычайно мощным: очень многие результаты, которые ранее подлежали доказательству, станут для него очевидным. Итак, система становится развивающейся — она перестраивается в ходе функционирования, причем перестройка касается не только ее информационной базы, но и известных системе доказательств теорем.

Перестройка доказательств будет происходить, во-первых, по причине улучшения самого алгоритма очевидности; во-вторых, в результате развития вычислительной техники (замена устаревших ЭВМ более мощными, способными с помощью прежнего алгоритма за ту же единицу времени осуществлять более длинные доказательства); и, наконец, в-третьих, вследствие появления новых обобщающих математических результатов. Общий результат тем и силен, что из него непосредственно вытекают построения, доказывающиеся ранее с помощью особых приемов. Последнее открывает очень интересную возможность (впервые появившуюся в истории дедуктивных наук) объективной оценки обобщающей силы математического результата. Нередко о математическом результате говорят: «Эта теорема весьма обща и глубока, поскольку имеет очень много следствий». Мы интуитивно чувствуем обобщенную силу теорем, но измерить ее не умеем. В системе, путь к созданию которой мы набрасываем, силу результата дедуктивного процесса можно будет измерять точно. Можно, например, предположить, что искомым «измерителем» окажется логарифмическая мера: логарифм отношения числа ячеек памяти, которые были заняты доказательствами всех теорем в данной области математики до этой теоремы, к числу ячеек, занятых для этой же цели после ее доказательства.

Из возможности измерения обобщающей силы дедуктивного результата возникает еще один интересный случай. Рассмотрим его на примере частного, но важного случая. В настоящее время существует различное отношение к работам в области «чистой» математики и к деятельност

математиков-программистов и «прикладников»: считается, что последняя носит более «приземленный» характер. Введение точной оценки «силы» результата лишит это утверждение основания: окажется возможным любые результаты сравнивать «на одних и тех же весах». Если усовершенствования операционной системы ЭВМ гораздо больше отразились на уменьшении общего количества слов, которое требуется для доказательств теорем в данной области, чем доказательство какой-либо новой (красивой и интересной) теоремы, то обобщающая сила первого результата, достигнутого умелым и изобретательным программированием, глубже, чем обобщающая сила второго результата, достигнутого в рамках абстрактной математики.

Практика свидетельствует о том, что предлагаемую автоматизированную систему дедуктивных построений можно реализовать уже на существующих машинах, если снабдить их хорошими пультами ввода — вывода информации, развитыми операционными системами, способными обеспечивать эффективное взаимодействие человека с машиной, и большой внешней памятью, достаточной для существенного расширения их информационной базы. Само же построение таких систем явится стимулом дальнейшего развития вычислительной техники и дедуктивной логики.

Работы по автоматизации процессов рассуждения имеют значение не только для математики (или физики), но и для любой науки и вообще для любого вида интеллектуальной деятельности. Поэтому можно с уверенностью полагать, что по мере развития вычислительной техники доля работ по программированию невычислительных дедуктивных построений будет все более возрастать за счет чисто вычислительных задач. На конструкцию машин, их внутренний язык и систему операций, на организацию связи человека и машины и машинной памяти и т. д. все большее влияние, вероятно, будут оказывать идеи, связанные с развитием такого рода систем. Вместе с тем возможности этих систем будут постепенно охватывать все более широкий круг проблем, выходящих далеко за пределы собственно математических задач.

В построении и организации соответствующих вычислительных систем весьма перспективной является идея использования большой машины в комплексе с малыми машинами, являющимися одновременно ее «интеллектуальными» пультами. Дело в том,что в будущем дедуктивные построения будут рождаться, как правило, в процессе интенсивного многошагового взаимодействия человека с машиной. Поскольку рентабельно лишь коллективное пользование большой машиной, а в этом случае начинают сильно сказываться технические ограничения на коммуникационные связи, то более рациональным оказывается размещение части общей «интеллектуальности» соответствующей машинной системы на ее периферии — на малых машинах, предназначенных для решения задачи диалога с человеком, который непосредственно работает с системой. Тогда центральная часть системы будет использоваться в основном как хранилище информации с быстрым выбором из памяти и быстрым проведением выкладок, связанных с построением доказательств. Расшифровка же высказываний человека, выбор направления перебора и т. д. будут осуществляться непосредственно на соответствующем «интеллектуальном» пульте.

СПИСОК ЛИТЕРАТУРЫ

1. *Глушков В. М.* Некоторые проблемы теории автоматов и искусственного интеллекта // Кибернетика.— 1970.— № 2.

2. *Глушков В. М., Капитонова Ю. В.* Автоматизация поиска доказательств теорем математических теорий и интеллектуальные машины // Кибернетика.—1972.— № 5.

3. *Ануфриев Ф. В., Костяков В. М., Малашонок А. И.* Алгоритм и машинный эксперимент поиска доказательств теорем в исчислении высказываний // Кибернетика.— 1972.— № 5.

4. *Глушков В. М.* Строение локальных и компактных групп и пятая проблема Гильберта // Успехи математических наук.— 1957.— 12, № 2.

О НЕКОТОРЫХ ПЕРСПЕКТИВАХ РАЗВИТИЯ И ПРИМЕНЕНИЯ ОБУЧАЮЩИХ МАШИН

(Известия вузов. Радиотехника.— 1963.— № 4)

Прежде чем говорить об обучающихся машинах, необходимо сказать несколько слов о таком важнейшем понятии кибернетики, как понятие обратной связи. При управлении сложными процессами, протекающими в условиях, всей совокупности которых нельзя заранее предвидеть, от действительности обратной связи в значительной мере зависит эффективность управления. Действительно, перед тем, как выработать то или иное целесообразное управляющее воздействие, необходимо получить информацию об условиях, в которых находится объект управления. После же посылки каждого очередного управляющего сигнала информация о его воздействии на объект управления представляет собой ту основу, на которой вырабатываются решения о характере последующих управляющих воздействий. Получение информации, о которой здесь идет речь, и составляет сущность обратной связи.

Если объем информации, поступающей по каналу обратной связи, недостаточен или эта информация поступает с большим запаздыванием, то эффективность процесса управления может резко снизиться. Представим себе, например, сталевара, который получает лишь весьма отрывочные данные о составе закладываемых в мартеновскую печь материалов. Пусть далее данные анализа выплавленной им стали становятся известными ему лишь через полгода после плавки. Ясно, что при таких условиях о качественной плавке не может быть и речи, поскольку эффективное управление процессом плавки невозможно.

Само собой разумеется, что описанная ситуация в действительности не имеет места. Есть, однако, один важный тип управляющих процессоров, в которых эта ситуация становится реальной. Речь идет о процессе обучения, особенно о процессе обучения студентов в вузе. Действительно, с абстрактно-кибернетических позиций роль преподавателя в учебном процессе сводится к управлению процессом приобретения знаний студентами, и в этом смысле она в какой-то мере аналогична роли сталевара в управлении процессом выплавки стали в нашем примере.

Характерной чертой процесса обучения в вузе является дальнейшее уменьшение (по сравнению со средней школой) роли канала обратной связи.

Действительно, в средней школе вследствие систематического опроса и проверки домашних заданий учитель имеет возможность более или менее регулярно, с запаздыванием всего лишь на одну-две недели, получать информацию (правда, далеко не полную) о степени усвоения материала каждым учащимся. В вузе же (особенно на старших курсах) соответствующая информация (притом, в уменьшенной дозе) часто поступает с запаздыванием на полгода или даже на год — во время очередной экзаменационной сессии.

Между тем, опыт показывает, что даже в средней школе при обычно принятых методах обучения действенность канала обратной связи (от учащихся к педагогу) оставляет желать лучшего и приводит к заметному снижению эффективности обучения. Об этом свидетельствует хотя бы тот общественный факт, что с помощью индивидуальных занятий с каждым отдельным учащимся можно значительно поднять эффективность обучения. Отметим, что индивидуальный подход важен не только для плохих, но и для хороших учеников, которые за то же время способны усвоить гораздо больше материала, чем им обычно сообщается в школе. Эффективность индивидуальных занятий в первую очередь определяется именно действенностью канала обратной связи: работая с небольшим числом учеников, преподаватель может немедленно выявлять и соответствующим образом реагировать на все возникающие у них затруднения с учетом индивидуальных особенностей каждого ученика.

Однако, к сожалению, «пропускная способность» человеческого мозга, а следовательно, и производительность труда педагога далеко не безграничны. Если преподаватель может вести эффективные индивидуальные занятия одновременно с двумя-тремя учащимися, то он заведомо лишен этой возможности, когда число учащихся выражается несколькими десятками, не говоря уже о сотнях или тысячах.

Сразу оговоримся, что речь здесь идет о преподавателе, который работает традиционными методами и не пользуется достижениями современной техники. При отказе же от этого ограничения положение коренным образом меняется: увеличивая техническую оснащенность процесса обучения, можно резко повысить производительность труда педагога, снять ограничения, накладываемые относительно малой пропускной способностью человеческого мозга.

Уже сегодня в учебном процессе с успехом используются многие технические средства, такие как кино, телевидение, звукозапись и т. п. Однако до последнего времени все эти средства были направлены в сторону увеличения возможностей лишь одного канала — прямой связи (от преподавателя к учащимся). С помощью радио и телевидения лекция, которую ранее могли слушать в лучшем случае лишь несколько сотен человек одновременно, становится доступной многомиллионной аудитории. Использование звукозаписи позволяет учащимся многократно (столько раз, сколько нужно) прослушивать записанную лекцию, что имеет большое значение, например, при изучении иностранных языков.

Нетрудно понять, что применение описанных средств во много раз увеличивает пропускную способность канала прямой связи. Что же касается канала обратной связи (от учащихся к преподавателю), то здесь дело обстоит сложнее. Действительно, в этом случае далеко не достаточно очевидно, организовать простую передачу данных от учащихся к преподавателю. Главное состоит в переработке этих данных в мозгу преподавателя, а этот участок до самого последнего времени оставался недоступным для автоматизации. В результате учебный процесс в целом не мог быть в сколько-нибудь полной мере автоматизирован, и, как следствие этого, производительность труда педагога оставалась практически на том же уровне, что и много десятков лет назад.

В настоящее время положение изменилось: успехи кибернетики и вычислительной техники привели к созданию обучающих машин, позволяющих организовать действенный двусторонний контакт преподавателя с многими десятками (а в перспективе — и со многими тысячами) учеников. Под действенностью контакта здесь понимается то, что каждый из взаимо

действующих с обучающей машиной учащихся находится в тех же условиях, что и учащийся, с которым преподаватель проводит индивидуальное занятие.

Подчеркнем, что использование обучающих машин (во всяком случае на современном этапе их развития) не только не уменьшает, а скорее увеличивает роль преподавателя в организации учебного процесса. Однако характер деятельности педагога при этом коренным образом меняется: основное внимание переносится не на само занятие, а на подготовку к нему. Подготовка же занятия в этих условиях представляет собой ни что иное, как создание соответствующего раздела так называемого **программированного учебника.**

Программированным называется учебник, снабженный столь подробными и точными указаниями о порядке его применения, что, пользуясь этими указаниями, человек, не знакомый с излагаемым предметом, мог бы провести достаточно квалифицированное занятие с любым учеником по заданной теме и вполне объективно оценить знания, приобретенные учеником во время этого занятия.

На первый взгляд задача создания подобного учебника может показаться неразрешимой. В действительности, однако, дело обстоит далеко не так безнадежно. Несмотря на то, что задача создания достаточно хороших программированных учебников весьма трудна, она, тем не менее, может быть успешно решена. Более того, педагогическая практика фактически уже сделала (хотя и с несколько иными целями) ряд шагов на пути решения этой задачи. Действительно, сопроводив каждый раздел учебника контрольными вопросами и задачами, мы делаем определенный шаг в направлении превращения его в программированный учебник. Изучив характерные ошибки, которые допускаются при решении контрольных задач, можно сделать еще один шаг, поместив в учебник указания, какие разделы нужно снова прочитать и какие новые контрольные задачи следует решить в случае той или иной конкретной ошибки. Следующий шаг состоит в разработке точных указаний о том, насколько должна быть снижена оценка при тех или иных ошибках. В случае, если учащийся затрудняется решить какую-то контрольную задачу, в учебнике следует предусмотреть возможность сообщения ему некоторых дополнительных наводящих разъяснений и вопросов. При этом также должны быть предусмотрены размеры соответствующего снижения оценок.

Продолжая работу подобным образом, мы в конце концов придем к созданию программированного учебника. Имея же такой учебник, можно построить машину, которая будет последовательно (следуя приведенным в учебнике указаниям) выдавать учащемуся необходимый материал для обучения, задавать ему вопросы, оценивать ответы и т. п. Это и будет требуемая обучающая машина, осуществляющая в соответствии с заданной программой индивидуальное обучение того или иного числа учеников.

Легко понять, что качество обучения с помощью обучающей машины всецело определяется качеством заложенного в нее программированного учебника. При этом ответственность, возлагаемая на составителя программированного учебника, гораздо выше, чем в случае обычных учебников. Если дефекты обычного учебника могут быть в той или иной мере исправлены преподавателем во время занятий, то для программированных учебников, закладываемых в обучающие машины, подобная возможность на современном этапе развития обучающих машин фактически исключается. Составитель программированного учебника должен быть не только специалистом излагаемого в нем материала, но и быть достаточно иску-

шенным в тонкостях самого педагогического процесса (знать характерные ошибки и затруднения учащихся, умело ставить наводящие вопросы и т. п.).

Разумеется, создание достаточно хороших программированных учебников потребует огромной затраты квалифицированного труда. Поэтому необходимо организовать параллельную работу (на базе уже существующих, непрограммированных учебников) большого числа опытных педагогов по программированию отдельных разделов соответствующих курсов. Подчеркнем, что затраты, о которых здесь идет речь, носят разовый характер и поэтому при широком внедрении обучающих машин должны быстро окупиться. В дальнейшем потребуется лишь работа по совершенствованию уже созданных программированных учебников, которая, несомненно, будет гораздо менее трудоемкой и, во всяком случае, не потребует того огромного числа людей, которые заняты педагогической работой в настоящее время. Труд педагога при этом заметно облегчится и в значительно большей мере, чем сейчас, будет носить творческий, научный характер.

В дополнение к сказанному отметим, что работа по созданию программированных учебников имеет и самостоятельную ценность (безотносительно к использованию обучающих машин), поскольку эти учебники (с небольшими видоизменениями) могут заметно поднять эффективность заочного образования и вообще всех видов самостоятельной работы учащихся. При существующих в нашей стране масштабах заочного образования одно лишь это может быть достаточно серьезным доводом в пользу организации систематической работы по созданию программированных учебников. Нельзя не видеть также и того, что такое мероприятие существенно повысит научный уровень всей учебно-методической работы как в средней школе, так и в вузах.

Прежде чем от чисто педагогических аспектов проблемы внедрения обучающих машин перейти к техническому аспекту этой проблемы, предостережем от опасности превращения обучающих машин в «натаскивающие». Эта опасность кроется именно в педагогическом, а не техническом аспекте и может быть устранена на этапе составления соответствующих программированных учебников. Изложение материала и выбор контрольных задач должны быть осуществлены так, чтобы развивать у учащихся творческий подход к изучаемому предмету, а не зазубривание тех или иных сведений о предмете, или приемов решения тех или иных задач. Набор контрольных вопросов и задач должен быть достаточно велик, чтобы исключить возможность простого заучивания всех правильных ответов без понимания сути изучаемого предмета.

Переходя к техническому аспекту проблемы развития и применения обучающих машин, прежде всего отметим, что обучающие машины можно разделить на так называемые универсальные и специализированные. Универсальные машины отличаются тем, что они способны реализовать любые правила обучения, которые только пожелает ввести составитель программированного учебника. У специализированных машин соответствующие возможности в той или иной мере ограничены.

Каждая универсальная обучающая машина должна иметь в своем составе устройство, способное выполнять (в зависимости от введенной в него программы) любые преобразования информации, которые можно определить с помощью конечных систем правил (произвольной природы). В настоящее время техника располагает только одним видом устройств с требуемыми свойствами — это универсальные электронные цифровые машины.

Таким образом, универсальные обучающие машины должны строиться на базе универсальных электронных цифровых машин.

Чтобы превратить универсальную электронную цифровую машину в обучающую, необходимо снабдить ее специальными пультами (по одному на каждого учащегося). В задачу этих пультов будет входить ввод и вывод информации, которой обмениваются между собой обучающая машина и учащиеся. Можно предложить много вариантов конструкций подобных обучающих пультов. В простейшем случае обучающий пульт представляет собой обычную пишущую машинку, способную приводиться в действие как человеком (учащимся), так и обучающей машиной. В первом случае осуществляется передача информации от учащегося к машине (обратная связь), во втором — от машины к учащемуся (прямая связь).

С целью избежания хранения в памяти обучающей машины всего учебного материала и экономии времени на вывод информации, каждого учащегося можно снабдить обычным (непрограммированным) учебником, а также сборником контрольных вопросов и задач. В память же машины заносится собственно программная часть программированного учебника (указания о порядке изучения материала, правила оценки ответов и т. п.). В таком случае обучающая машина будет выдавать (через пишущие машинки на индивидуальных пультах у каждого учащегося) лишь короткие указания типа: «Ознакомиться с таким-то разделом учебника», «Решить такую-то задачу» и др., а также наводящие вопросы и оценки получаемых ответов.

Использование наряду с обучающей машиной обычных учебников имеет еще и то преимущество, что позволяет пользоваться одним пультом нескольким учащимся: пока один из них связывается через пульт с машиной, остальные могут быть заняты изучением предложенного им материала или решением задач. Можно, разумеется, вместо учебников (или наряду с ними) снабдить пульты кинопроекторами, магнитофонами и наборами лент, на которых записан полный цикл лекций по изучаемому курсу. В этом случае каждый учащийся должен иметь возможность воспроизвести для себя требуемую лекцию или любую ее часть столько раз, сколько это ему потребуется.

Использование заснятых на пленку лекций (вместо учебников) имеет то преимущество, что такие лекции можно довольно часто обновлять по мере развития соответствующей дисциплины или приобретения большого опыта в ее преподавании. Ради этого часто можно мириться даже с усложнением (и соответственно с удорожанием) обучающих пультов.

Ответы учащихся на вопросы, задаваемые обучающей машиной, а также, если возникает необходимость, и их встречные вопросы машине при печатании на пульте автоматически вводятся в обучающую машину. Наиболее трудной проблемой является при этом проблема «понимания» обучающей машиной информации, получаемой ею от учащихся. Указанную проблему можно решить на нескольких различных уровнях, и степень совершенства ее решения в значительной мере предопределяет степень сложности соответствующей обучающей программы (обучающей машины).

Наиболее примитивным является уровень, основанный на принципе однозначности правильного ответа. В этом случае предполагается, что на каждый вопрос, заданный машиной, может быть один и только один правильный ответ. Обычно предполагается также, что все неправильные ответы объединены в конечное число классов (в простейшем случае в один класс), а к каждому такому классу заранее отнесены соответствующие снижения оценок (быть может, с предварительным заданием ряда наводя-

щих вопросов), а также указания в отношении разделов курса, которые надо снова проработать, и контрольных задач, которые следует решить дополнительно.

Круг задач с однозначно определенными ответами достаточно широк. Он включает в себя большинство математических задач (хотя и не все), а также инженерно-технических задач расчетного характера. Несколько хуже обстоит дело не с расчетными задачами, а с контрольными вопросами, на которые требуется давать устные ответы. Как известно, на любом языке одна и та же мысль может быть выражена многими различными способами. Это усложняет задачу распознавания правильного ответа, если разумеется, не требовать заучивания наизусть точных формулировок.

Для упрощения данного процесса в таком случае прибегают к специальным формам построения вопросов. Наиболее простая из них предусматривает такое построение вопроса, чтобы он допускал односложный ответ в виде слов «да» или «нет». Один из двух указанных ответов будет правильным, а второй ошибочным. Несколько более сложная форма задания вопросов, сохраняющая принцип однозначности правильных ответов, состоит в том, что учащимся предлагается сделать выбор между несколькими сообщаемыми ему ответами на поставленный вопрос. Лишь один из этих ответов является правильным, и для того, чтобы успешно справится с заданием, учащемуся достаточно указать номер правильного ответа. Зная характерные ошибки, совершаемые учащимися, составитель программированного учебника может использовать их для составления неправильных ответов, что позволит предупредить повторение подобных ошибок после окончания курса обучения.

Важно отметить, что при соблюдении условия однозначности правильных ответов, как правило, нет необходимости строить обучающие машины на базе универсальных электронных цифровых машин. В этом случае обычно оказывается возможным использовать гораздо более простые и дешевые конструкции обучающих машин. В настоящее время на таких принципах построен и успешно эксплуатируется ряд относительно простых обучающих машин. Несмотря на простоту, эти машины оказываются достаточно эффективными и во многих случаях заметно сокращают сроки и улучшают качество обучения.

Вместе с тем нетрудно понять, что строить учебный процесс полностью на базе подобных простейших машин нельзя. Действительно, хорошо известно, что для целей контроля и управления процессом обучения наибольшую ценность представляет не сам по себе ответ на предложенную задачу, а метод, с помощью которого учащийся нашел этот ответ (правильный или неправильный). Лишь следя за ходом рассуждений учащегося в процессе решения задачи, преподаватель может составить себе достаточно ясное представление о степени усвоения материала, а тем более о развитии творческих способностей учащегося.

Но ведь основной целью хорошо поставленного обучения должно являться именно развитие творческих способностей, а не только простое накопление знаний и овладение определенными навыками в решении тех или иных стандартных задач. Подобная постановка проблемы вынуждает нас расширить понятие ответа, включив в него не только собственно ответ на предложенный вопрос (задачу), но обоснование этого ответа (решение задачи). В таком случае, как легко понять, об однозначности правильного ответа не может быть и речи. Действительно, при развитии у учащихся творческого подхода к изучаемому предмету нужно всячески поощрять оригинальные нестандартные приемы решения. В то же время

при соблюдении условия однозначности правильного ответа (а который теперь включается и решение) подобные решения будут неизбежно классифицироваться как неправильные.

Трудность, с которой мы встретились, ставит нас перед необходимостью перейти на более высокую степень организации обучающих машин, отказавшись от принципа однозначности правильного ответа. В этом случае множество всех ответов на заданный вопрос разбивается на два подмножества, состоящих соответственно из всех правильных и неправильных ответов. Оба эти подмножества могут, в свою очередь, разбиваться на еще более мелкие подмножества в соответствии с теми или иными принципами. Так, правильные ответы (вместе с решениями) могут различаться дополнительно по признакам точности, краткости, оригинальности, а неправильные — по характеру сделанных ошибок.

Наиболее простым является случай, когда множества правильных и неправильных ответов конечны. Этот случай при небольшом числе возможных ответов мало чем отличается от случая, когда выполняется условие однозначности правильных ответов, поскольку программа может быть построена по принципу простого перечисления всех этих ответов и сопоставления с ними соответствующих реакций обучающей машины.

Если же число возможных ответов на заданный вопрос становится большим, то необходимо разрабатывать правила, с помощью которых должно производиться распознавание правильных и неправильных ответов (а также различных их модификаций). Разрабатывая эти правила применительно к тому или иному конкретному способу записи ответов и вопросов, мы получаем некоторый абстрактный язык, пригодный для обмена информацией между обучающей машиной и учащимися. При этом, как и в обычной разговорной человеческой речи, существует вообще много различных способов для выражения одной и той же мысли, однако с помощью соответствующей системы правил, введенных в программы, машина как бы «улавливает смысл» фраз на рассматриваемом языке, отвлекаясь (в той мере, в какой это возможно) от их конкретного языкового оформления.

Подобные абстрактные языки (называемые также информационными) можно разработать применительно к той или иной конкретной области знания. Примером специализированного информационного языка может служить, скажем, язык алгебраических формул. Правила эквивалентных преобразований формул позволяют оперировать фактически классами эквивалентных друг другу формул, а не отдельными формулами. В результате если, допустим, ответом на заданный вопрос была алгебраическая формула $(a + b)^2$, то машина может самостоятельно отождествить ее с формулами $a^2 + 2ab + b^2$, $(a + b)(a + b)$ и др.

Более трудной является задача создания информационного языка для записи доказательств в дисциплинах математического характера. Разработанные в математике логические языки (называемые обычно исчислениями) в большинстве случаев непригодны для применения в обучающих машинах, поскольку они весьма мало похожи на обычно используемые способы записи доказательств. Предстоит еще большая работа по превращению этих сугубо теоретических языков в языки, пригодные для массового практического употребления. Подобные языки создаются сейчас для ряда ограниченных областей математики и других точных наук. Как пример отметим, что еще в 1958 г. автор провел успешный опыт применения одного специального языка для автоматической проверки (с помощью универсальной электронной цифровой машины) правильности доказательств в одной новой алгебраической теории.

Отметим, что проблемы, которые должна решать обучающая машина в случае неоднозначности даваемых учащимися ответов, аналогичны хорошо известной в кибернетике проблеме, называемой проблемой распознавания образов. Действительно, в задачу машины входит разделение множества всех возможных ответов на два подмножества (правильных и неправильных ответов). В случае же распознавания (зрительных) образов машина должна разделить все возможные рисунки на два класса, а именно на рисунки, изображающие некоторый объект, и на рисунки, не изображающие этот объект.

Подобная аналогия наводит на мысль применения для целей построения обучающих машин успешно используемых в случае распознающих машин идей самоорганизации и самообучения. Прямое перенесение указанных идей на интересующий нас случай оказывается невозможным. Однако, как показал автор, используя несколько иные принципы, можно построить самоорганизующиеся системы, способные настраиваться на распознавание языковых образов.

Эти принципы были реализованы в виде программ, позволяющей организовать процесс обучения машины распознаванию смысла фраз на русском языке. Подробное описание, связанное с организацией программы и самого процесса обучения, можно найти в совместной работе автора с Н. М. Грищенко и А. А. Стогнием [1]. Отметим лишь, что, как показали эксперименты, обучение машины указанной задаче внешне ничем не отличается от обучения человека. Очень важно то, что машина обучается правильно распознавать смысл тех фраз, которые сообщались ей в период обучения, но и таких фраз, которые ей ранее никогда не сообщались.

Значение проведенных экспериментов для совершенствования обучающих машин легко понять. Эти эксперименты показывают, что в качестве языка для общения учащихся с обучающей машиной может быть использован обычный русский (или любой другой «человеческий») язык. При этом нет безусловной необходимости в предварительном изучении и точном формулировании всех не только грамматических, но и синтаксических правил языка: эти правила (а число их огромно) может найти сама машина в процессе ее совместной работы с учителем.

Опираясь на уже имеющийся опыт, можно высказать уверенность, что работа в области создания машинных языков, с одной стороны, и работа в области теории самоорганизующихся систем, с другой, рано или поздно приведут к тому, что машина сможет свободно оперировать обычными «человеческими» языками. В этом и только в этом случае учебный процесс может быть полностью ориентирован на использование обучающих машин.

Но и при самых совершенных обучающих машинах роль преподавателя в системе образования по-прежнему остается высокой (хотя численность преподавателей может заметно уменьшиться). Не говоря уже о повседневной работе по корректировке и совершенствованию закладываемых в машину программ, потребуется немалая работа по созданию новых учебных курсов по принципиально новым разделам науки. Не следует также забывать, что воспитательная работа представляет собою органическую составную часть системы образования, а одним из самых могучих средств воспитания является, как известно, личный пример преподавателя.

И, наконец, следует помнить об экономических факторах. Дело в том, что достаточно совершенные обучающие машины представляют собой сегодня сложные и дорогие устройства, нуждающиеся в высококвалифицированном персонале для их нормальной эксплуатации. Потребуется еще

известный период совершенствования технологии изготовления и увеличения надежности дискретной кибернетической техники, прежде чем универсальные обучающие машины придут в каждый вуз, и тем более, в каждую школу. Большая роль в ускорении сроков внедрения совершенных обучающих машин будет принадлежать микроэлектронике, твердым схемам и другим принципиально новым способам построения схем электронных цифровых машин.

Значительную роль в создании совершенных обучающих машин призваны сыграть работы по читающим автоматам. При этом речь здесь идет не только о специализированных автоматах для ввода в обучающие машины рукописных или машинописных текстов, но и об универсальных читающих автоматах, способных воспринимать и расшифровывать произвольную зрительную информацию. Экспериментальный (пока еще несовершенный) образец подобного универсального автомата создан и успешно работает в Институте кибернетики АН УССР [2]. При дальнейшем усовершенствовании таких устройств обучающие машины как бы обретут зрение. В результате станет возможным воспринимать информацию в виде рисунков, чертежей и т. д., не прибегая к специальным ухищрениям для ее ввода (световые карандаши, координатные доски и др.). Важное значение для совершенствования обучающих машин будут иметь также ведущиеся в настоящее время работы по непосредственному вводу информации в машину голосом и соответствующей системы вывода информации.

Из сказанного ясно, что для создания и широкого внедрения обучающих машин, действующих на достаточно высоком уровне, потребуется еще большая работа во многих областях кибернетики. Однако было бы неправильным делать вывод, что вопрос о широком применении обучающих машин в педагогической практике представляет собой дело далекого будущего, так как уже на существующей технической базе можно на короткое время наладить выпуск простых и достаточно дешевых обучающих машин. При должной организации учебно-методической работы такие машины могут найти широкое применение во всех звеньях народного образования (и, в первую очередь, в вузах) в самое ближайшее время. Не подменяя преподавателя, они могут заметно облегчить его труд, и, что самое главное, с их помощью можно существенно поднять качество обучения.

В будущем же, по мере удешевления средств электронной вычислительной техники, подобные простые машины могут быть постепенно заменены более совершенными системами автоматизации обучения. Попутно при внедрении средств автоматики в учебный процесс на каком-то этапе может быть решена также задача автоматизации учета, статистики и отчетности, связанных с системой образования.

Уже сейчас вузы, располагающие универсальными электронными цифровыми машинами, могут дооборудовать их специальными пультами и использовать для целей программированного обучения. Важно отметить, что одна современная универсальная электронная цифровая машина может одновременно работать с несколькими десятками студентов. Разумеется, при этом необходимо иметь соответствующее число пультов, а также буферную память и другие устройства для сопряжения пультов с машиной. Особенно перспективно использование универсальных электронных цифровых машин для обучения программированию. Задача здесь облегчается, так как язык программирования является для машины «родным языком», что очень облегчает составление соответствующих обучающих программ. В то же время вследствие большой потребности в программистах, авто-

матизация обучения программированию является одной из наиболее актуальных задач.

Будучи могучим средством повышения производительности труда огромной армии работников педагогического фронта, автоматизация учебного процесса на базе внедрения обучающих машин поможет в короткие сроки решить многие задачи по дальнейшему совершенствованию системы образования в нашей стране.

СПИСОК ЛИТЕРАТУРЫ

1. *Глушков В. М., Грищенко Н. М., Стогний А. А.* Алгоритм распознавания осмысленных предложений // Принципы построения самообучающихся систем.— Киев : ГИТЛ, 1962.— 19.— С. 26.

2. *Глушков В. М., Ковалевский В. А., Рыбак В. И.* Универсальная установка для исследования алгоритмов распознавания изображений // Там же,— 63,— С. 72.

ЭЛЕКТРОННЫЕ МАШИНЫ И АВТОМАТИЗАЦИЯ УМСТВЕННОГО ТРУДА

(Будущее науки, М. : Знание.— 1966)

Электронные вычислительные машины — одно из наиболее удивительных созданий науки и техники XX в. Их вполне можно поставить в один ряд с практическим использованием атомной энергии или началом освоения космоса. Правда, рождение электронных вычислительных машин было не таким эффектным, но с течением времени они стали завоевывать все новые и новые позиции. Теперь все большее число ученых склоняются к мысли, что в конечном счете появление электронных вычислительных машин сыграет для человечества не меньшую, а по-видимому, даже большую роль, чем атомная энергия или космические полеты. Это объясняется тем, что электронные вычислительные машины — универсальные преобразователи информации, а с преобразованием информации человек сталкивается всегда в любой сфере своей деятельности. По сути, именно преобразованием информации занимается и переводчик, и экономист-плановик, и математик, и даже поэт. Преобразование информации — это и содержание того, что мы называем умственным трудом человека. А так как и во время, когда человек выполняет чисто физическую работу, мозг его работает, координируя движения рук и ног, то, по сути, нет ни одного участка деятельности человека, не связанного с преобразованием информации.

Именно вследствие такой неограниченной области применения электронных вычислительных машин их научное и техническое значение будет изо дня в день расти.

Почему мы электронные вычислительные машины называем универсальными преобразователями информации? На первый взгляд, здесь как будто есть противоречие: электронные машины имеют дело с информацией только одной природы — числовой, тогда как в перечисленных выше процессах преобразования информации участвует самая разнообразная информация. Действительно, информация, которую человек воспринимает,— это и звуковая, причем не только осмысленная, но и различного рода шумы, в конце концов музыка, и зрительная — все богатство форм и красок внешнего мира. А электронные вычислительные машины оперируют только числами.

Но противоречие это лишь кажущееся. Нетрудно показать, что числовой способ задания информации является в некотором смысле универсальным, т. е. любую информацию путем сравнительно несложных преобразований можно привести к числовому виду. Более того, уже разрабатываются информационные устройства, позволяющие любой вид информации преобразовать в числовую форму, и наоборот. Правда, эти аппараты сейчас еще не всегда надежны, обладают недостаточной скоростью и т. д. Но они существуют.

Сомнение в универсальности вычислительных машин как преобразователей информации может вызывать еще и то, что правила преобразования

информации различной природы качественно различны. Одни правила используются в математике при решении вычислительных задач и совсем другие — при доказательстве теорем.

Но и здесь мы подходим к одному из фундаментальных фактов, который был установлен математической логикой еще в домашинный период, но значение которого для человечества стало ясно только после того, как появились электронно-вычислительные машины.

Мы не удивляемся, что множество разнообразных предметов, окружающих нас, в конце концов состоит из одних и тех же элементарных частиц в разных комбинациях. Электроны и протоны одинаковы везде, но тем не менее сочетания их в атомах и молекулах создают совершенно различные тела.

А почему бы информации быть в этом смысле исключением? Почему не может быть «информационных атомов», атомов преобразования информации, на которые можно разложить любые правила ее преобразования? Оказывается, это сделать можно. Можно выделить небольшое число типовых правил — «атомов», с помощью которых можно представить, или, как выражаются в электронно-вычислительной технике, запрограммировать любые правила, если только эти правила познаны и точно описаны. Природа этих правил роли уже не играет. Это могут быть правила грамматики, математики, стихосложения, музыкального творчества, экономического анализа и т. д. Если эти правила познаны и точно описаны, их можно разложить на некоторые элементарные правила.

Электронная вычислительная машина впервые в истории человечества вместила в себя весь набор элементарных правил преобразования информации и имеет принципиальную возможность выполнять по этим правилам любые действия в заданной последовательности. Это и делает электронные вычислительные машины универсальными преобразователями информации, универсальным средством автоматизации не только физического, но и умственного труда человека, причем умственного труда достаточно высокой квалификации.

По сути, на наших глазах происходит вторая техническая революция. Первая такая революция, затронувшая область физических усилий, была связана с созданием двигателя, умножившего физическую мощь человечества. Теперь же мы являемся свидетелями рождения универсальных автоматов, которые помогут неограниченно увеличить интеллектуальную мощь человечества.

Разумеется, одно дело — принципиальная возможность и другое дело — ее практическая реализация. Чтобы действительно использовать все огромные возможности, уже заложенные в современных электронных вычислительных машинах, не говоря уже о машинах будущего, необходимо изучить правила, по которым человек преобразует информацию в той или иной сфере приложения своего интеллекта, в той или иной сфере умственной деятельности. А это задача колоссальной сложности.

Если позволено мне будет употребить такое сравнение, вычислительная машина подобна моагу только что родившегося ребенка, который нужно «начинить» соответствующей информацией. Потенциально ребенок может обладать такими возможностями, что из него может вырасти Ньютон или Лобачевский. Но необходима огромная работа, чтобы сообщить ему всю совокупность знаний, всю необходимую информацию, можно сказать, необходимые программы работы — программы очень высокого уровня, взаимодействие которых мы в полной мере еще ясно себе не представляем.

Нужно иметь в виду, что принципы, с помощью которых преобразование информации раскладывается на отдельные элементарные акты в мозгу человека, имеют только внешнее сходство с принципами, реализующимися в электронных вычислительных машинах. Тем не менее внешний конечный результат один и тот же: как мозг человека, так и электронные вычислительные машины — универсальные преобразователи информации, хотя построенные совершенно по-разному.

И теперь встает следующий, не менее важный и интересный вопрос. Если перед человечеством открываются столь широкие горизонты машинной переработки информации, облегчения умственного труда, а для реализации этих перспектив нужно будет проделать очень большую работу, то куда же следует направить основные усилия в настоящее время? Какие задачи в первую очередь должны интересовать человечество, науку? Что в первую очередь надо передать машинам, в какой области призвать их на помощь?

Можно, скажем, пытаться решить задачу машинной переработки информации в применении к автоматизации игры в шахматы: изучить правила, по которым гроссмейстер оценивает позиции, разложить их на элементарные правила и в конце концов получить программу. Но человечество еще не испытывает особого ущерба оттого, что люди играют в шахматы по-старому, без использования электронно-вычислительной техники. В то же время существуют такие области человеческой деятельности, где уже сегодня дальнейшее развитие без использования такой техники невозможно. Это те области, где количество информации, которое получает работающий в них человек, начинает превосходить его возможности. Возникает информационный затор: информация скапливается и человек не успевает ее перерабатывать. Тогда он прибегает к какому-нибудь качественному методу, методу качественных оценок и выполняет соответствующую работу хуже, с большим количеством ошибок.

Мы как-то не привыкли задумываться над тем, что не только физические, но и умственные возможности человека ограничены. Человека нельзя заставить мыслить все с большей и большей скоростью, не изменив существенно биологической природы его мозга. Вряд ли на протяжении ближайшего столетия появится человек, который способен будет запомнить наизусть все книги, хранящиеся в Библиотеке СССР имени В. И. Ленина, или выучить наизусть в течение часа «Большую советскую энциклопедию». В области умственной деятельности человек имеет определенный предел производительности труда, и когда к нему предъявляются требования, превосходящие этот предел, он не может их выполнить, так же как землекоп не может выкопать лопатой котлован для крупной гидроэлектростанции.

Но то, чего не может сделать человек как объект биологический, он может делать и успешно делает как объект социальный. Человеческое общество уже нашло способы штурмовать эти вершины информации. Изобретение книгопечатания, например, тоже было ответом человечества на увеличение количества информации. Благодаря ему человек как социальный объект восполняет свои недостатки как объекта биологического. В настоящее время эти возможности неограниченно умножились в связи с применением электронных вычислительных машин.

Таким образом, усилия ученых, работающих в области применения электронных вычислительных машин, нужно направить в первую очередь в те области человеческой деятельности, где уже сегодня ощущается информационный затор и где он будет испытываться в ближайшем будущем.

Надо сказать, что само появление электронных вычислительных машин было вызвано именно образованием такого информационного затора. В связи с развитием новых областей техники в вычислительной математике появились задачи, которые при старых методах счета потребовали бы десятков, сотен, а может быть и тысяч лет человеческого труда. Поэтому противоречие, создавшееся в результате возникновения такого информационного затора, было разрешено при помощи электронных вычислительных машин.

Первая задача, которая встает перед нами,— это задача автоматизации планирования, управления экономикой и учетом. Проблемы, возникающие в системе планирования, грандиозны. Наша промышленность, народное хозяйство непрерывно растут и управлять этим хозяйством становится все сложнее.

Например, перед Госпланом УССР стоит задача согласовать планы материально-технического снабжения с планом производства материалов. Для этого Госплан УССР должен получить исходную информацию в размере 100 млн чисел и произвести с этими числами примерно триллион операций. Что такое триллион операций? Если выполнять одну операцию в секунду, что человек заведомо не может делать на протяжении сколько-нибудь длительного времени, то для выполнения такого количества операций потребуется 30 тыс. лет. И это только для составления годового плана!

Ясно, что без автоматизации в сфере планирования невозможно решить те огромные задачи, которые встают перед страной. Если же применить электронную вычислительную машину, которая делает 100 тыс. операций в секунду (а эта скорость является в настоящее время довольно обычной), то подобная задача может быть решена за три-четыре месяца, а если применить несколько таких машин, то еще быстрее.

Таким образом, электронно-вычислительная техника дает возможность осуществлять гораздо более детальное планирование и наилучшим образом использовать те огромные преимущества, которые заложены в социалистической системе хозяйства.

Можно привести такой пример. С помощью электронных вычислительных машин теперь решаются многие транспортные задачи. Планы железнодорожных перевозок, составленные машинами, на 10—15 %, а план автомобильных перевозок чуть ли не в два раза экономичнее, чем планы, составленные «вручную».

В настоящее время электронные вычислительные машины используют в основном так называемые линейные модели, и математики приобрели большой опыт в применении их для планово-экономических расчетов. Сейчас возникает проблема построения динамических моделей, которые позволили бы найти наилучшие пути развития нашей экономики с целью обеспечить заданный уровень потребления на какой-то определенный период. Математики пытаются создавать методы, которые позволят решить такие задачи. И здесь в большом долгу перед наукой оказываются экономисты, которые должны разработать соответствующие экономические модели и дать возможность математикам применить эти методы решения.

Очень важная проблема в автоматизации планирования — автоматизация первичного учета. Те формы хранения информации, к которым мы привыкли, совершенно не удовлетворяют электронно-вычислительную технику и являются в наше время архаичными. Необходимо переходить к новым формам хранения первичной информации, удобным для последующей передачи на электронные вычислительные машины: на магнитной

ленте, перфокартах, специальных документных бланках. Сегодня каждый, кто занимается учетом и составлением документов, должен думать о том, как будет этот документ читать не только вышестоящая инстанция или контрагент, но и электронная вычислительная машина. Уже появляются такие системы, в которых подготовка первичных документов совмещается с передачей информации по каналам связи в электронные вычислительные машины. При такой организации все данные, содержащиеся, например, в какой-нибудь лежащей перед вами накладной, одновременно передаются в электронную вычислительную машину, которая их запомнит и использует в дальнейшем.

Предстоит еще огромная работа по увеличению парка электронных машин, занятых переработкой планово-экономической информации. Из этих машин нужно создать мощные вычислительные центры с современными каналами связи, чтобы можно было быстрее обмениваться информацией. Речь идет не только о составлении новых языков программирования, но и о перестройке всей системы ведения первичной документации, самой психологии людей, занятых в сфере учета и планирования.

Еще одно очень важное применение электронные вычислительные машины находят в комплексной автоматизации инженерно-технических расчетов. Здесь, вообще, машины используются довольно давно: значительная часть расчетов, необходимых при проектировании новой машины или изделия, уже производится не вручную, а электронной вычислительной машиной. Но мы здесь говорим о новой возможности комплексной автоматизации процессов проектирования и последующего изготовления машины или изделия, когда процесс проектирования становится органической составной частью самого производственного процесса.

Примером этого может служить одна из работ, выполненных Институтом кибернетики АН УССР вместе с Институтом автоматики (Киев) на одном из судостроительных заводов. Вычислительной машине задается исходная информация о корпусе судна, и она без всякого вмешательства человека по определенной заданной программе производит разверстку листов, из которых сваривается корпус судна, на плоскости и оптимальную раскладку без чертежей нужных деталей заданных размеров, а потом выдает магнитные ленты с программами. Если эту магнитную ленту заложить в соответствующий станок, то он будет вырезать газовым резаком нужные детали из большой полосы стали; если же газовый резак заменить чертежным оборудованием, по той же программе будут выполнены чертежи раскладки деталей в соответственно уменьшенном масштабе.

Все это дает экономию инженерно-технического труда, исчисляемую в 200 тыс. рублей в год. Еще большая экономия будет получена при сборке судна, потому что изготовление таким методом детали корпуса намного точнее выполненных старым способом и подгонка листов в процессе сборки корабля будет практически исключена.

Проектирование с помощью электронных вычислительных машин позволяет перейти от обычных методов к так называемым оптимальным, когда проверяется несколько вариантов решения той или иной технической проблемы и из них выбирается наилучший. Такая возможность в домашинный век, была, как правило, принципиально недоступна, так как построение оптимальных конструкций требовало огромных затрат умственного труда.

Остановимся еще на одной, особенно интересной задаче в области автоматизации проектирования — на автоматизации проектирования самих вычислительных машин.

В популярной литературе часто говорится о создании таких электронных вычислительных машин, которые будут производить себе подобных — будут способны, так сказать, к размножению. Сейчас это уже не праздная проблема, которая может интересовать человечество в будущем, а одна из актуальнейших задач сегодняшнего дня.

Существующие электронные вычислительные машины настолько сложны, что на их проектирование уходит очень много времени и конструктор никогда не бывает уверен, что он в данных условиях выбрал лучший вариант. В будущем же предстоит разрабатывать еще более совершенные электронные вычислительные машины. И вот возникает конкретная задача — поручить машине создавать, конструировать себе подобные или еще более совершенные машины.

Такие работы проводятся в Институте кибернетики АН УССР. Нельзя сказать, что они уже завершены, но развитие автоматизации в настоящее время значительно расширило возможности применения формальных методов синтеза для проектирования электронных вычислительных машин. Одна из основных проблем здесь состоит в создании знаковой системы — своего рода искусственного языка — для описания тех или иных уровней проектирования соответствующего объекта, например совершенно точного «языка» для описания блок-схемы машины. Такой язык сейчас рождается стихийно: один изображает блок-схему так, другой — иначе. Необходимо разработать точный способ представления структуры машины на данном уровне ее абстрактности, реализации блок-схемы, когда элементы машины еще не выбраны.

Ряд подобных языков в настоящее время создан и создается. После того, как структура машины записана на таком языке, к ней можно применить формулу минимизации и оптимизации схемы и получить такие машины, которые другим способом за короткое время построить не удается.

Правда, еще нельзя сказать, что проектирование электронных вычислительных машин полностью автоматизировано, но тем не менее определенный ряд очень трудоемких этапов уже пройден.

Следующая важная проблема, которая встает перед человечеством, — это автоматизация научных исследований. Уже сейчас математики, физики, механики, техники применяют в своей научной деятельности электронные вычислительные машины. Но пока эти вычислительные машины играют подсобную роль, т. е. используются для вычислений, а не для того, чтобы выполнять на машинах сам творческий процесс, а тем более — постановку задачи, что составляет суть науки. Об этом до самого последнего времени боялись даже мечтать, такая возможность казалась очень далекой.

Прежде всего подобно комплексной автоматизации инженерного проектирования можно уже говорить о комплексной автоматизации научных исследований в области экспериментальных наук. Машина может не просто производить те или иные расчеты, а выбирать объект исследования, например тот или иной физический прибор, присоединяться к этому прибору и самостоятельно проводить физический эксперимент, рассчитывать показания, обрабатывать их и выдавать готовый результат. Комплексная автоматизация исследований уже начинает осуществляться при решении таких задач, как анализ снимков звездного неба в астрономии или анализ следов частиц на снимках, полученных при фотографировании ядерных реакций. Машины изучают эти снимки, выбирают необходимый тип реакций, делают нужные замеры и в конце концов выдают обработанные данные.

Что касается теоретических наук, основанных на дедуктивных методах, то здесь возникает не менее интересная задача автоматизации самого процесса научного творчества. В области математики это прежде всего процесс доказательства трудных теорем.

Спрашивается, а нужно ли решать эту задачу сейчас? Так ли она актуальна в настоящее время?

Люди давно задумывались над тем, как ученый приходит к тому или иному результату. Особенно явно это видно в математике: здесь можно найти наиболее характерные примеры. Часто бывают случаи, когда сформулированная теорема в течение многих лет не доказывается, и большое число людей пытается найти подход к ее доказательству.

Анализируя процесс научного творчества, можно видеть, что человек пытается добиться решения на основе так называемой интуиции, аналогий с теми знаниями, которые он уже имеет, накопил в течение длительного времени, изучая специальную литературу, пробуя всевозможные варианты, исходя из уже решенных проблем и начальных данных. В конце концов у исследователя может возникнуть догадка, которая «соединит звенья цепи», и очень напряженная работа, длившаяся годами, завершится успехом.

Но так бывает не всегда. В той же математике существует очень много проблем, над которыми ломали голову выдающиеся математики, но тем не менее эти проблемы и сегодня остаются нерешенными. И получается, что человек двадцать лет программирует себя на решение той или иной задачи, потом еще несколько лет пытается ее решить, а в конце концов наступает старость и человек, который мог бы еще очень и очень многое сделать для науки, уже оказывается выбывшим из строя. Все то, что он мог бы еще написать, погибло.

Кроме того, если существуют проблемы, над которыми человеку надо думать непрерывно, например, в течение тридцати лет (а такие примеры сейчас не так уж редки), то могут существовать и такие проблемы, над которыми надо думать 200 лет, и человек не решит таких проблем только потому, что отведенный ему природный срок жизни не позволяет этого сделать.

Правда, человечество находит выход из этого положения за счет специализации, разложения проблемы на более мелкие подпроблемы, но где гарантия того, что слепые поиски приведут к разрешению проблемы? И сколько еще можно ждать счастливых случайных сочетаний, счастливых догадок исследователей? Темпы развития науки в настоящее время таковы, что, по-видимому, в ближайшем будущем человечество не сможет себе позволить такую роскошь, как ждать случая, дарованного провидением. Поэтому так важна задача автоматизации научного творчества.

На первый взгляд мы встречаемся с определенным противоречием. В математической логике доказано, что наряду с теориями, поддающимися разрешению путем создания универсальных программ, существуют и так называемые неразрешимые теории, о которых известно, что в их рамках нельзя сформулировать алгоритмы программ, которые бы доказывали или опровергали данную теорему. Казалось бы, это воздвигает предел применению электронных вычислительных машин.

Но оказывается, что по сути проблема теоретической алгоритмической разрешимости той или иной теории и проблема практической ее разрешимости являются полярно противоположными. Если изучить весь ход мыслей научного работника, все переходы, которые он использует, то можно прийти к заключению, что здесь действует сравнительно небольшое число

правил, которые и помогают ему разрешить определенный круг задач рассматриваемой теории. Таким образом, задача ставится по другому: найти не универсальный алгоритм, решающий все проблемы в данной области, а практически функционирующие алгоритмы, которые работали бы так же или лучше, чем математик, работающий в этой области. Поэтому есть надежда, что даже если не все практические правила, которыми пользуется человек в процессе доказательства той или иной теоремы, будут вложены в электронную вычислительную машину, она все равно сможет работать быстрее, чем человек.

Может оказаться, что в разрешимой теории, где уже построен алгоритм, решающий все проблемы, в действительности практически пользоваться этим алгоритмом нельзя, поскольку даже для вычислительной машины этот алгоритм оказывается громоздким; и наоборот, в неразрешимой теории, где доказана невозможность существования такого алгоритма, может быть достаточно простой алгоритм, охватывающий именно ту часть теории, которая в настоящее время интересна для человечества.

Если мы поставим сейчас задачу построить не такой алгоритм, который может доказать или опровергнуть все мыслимые теоремы в данной области, а такой, чтобы доказать теоремы, которые возникнут перед человечеством на протяжении XXI в., то данный алгоритм мы найти не сможем.

Правда, если даже сегодня удалось доказать те маленькие теоремы, на которые пытается делить доказательства больших теорем математическая логика, то такие доказательства трудно читать, и человеку придется преодолеть немало трудностей, чтобы перевести их на обычный русский, английский, немецкий или другой язык, на язык предикатов. Возникает вопрос: а не нужно ли построить такой язык, который был бы в какой-то мере подобен, скажем, русскому языку, не содержал бы меньше возможностей разных способов выражения одной и той же мысли, был бы более определенным и точным, использовал как можно меньше терминов? Для этого нужно построить алгоритм, который будет доказывать теоремы и выдавать эти доказательства в таком виде, чтобы их можно было читать и публиковать.

В настоящее время попытки построения подобных программ делаются в целом ряде научных коллективов. Есть некоторые успехи, есть неудачи, показавшие, что здесь предстоит преодолеть очень большие трудности. Во всяком случае, ясно, что мы уже приступили к практическому решению этой проблемы.

Оказывается, что наиболее трудная ее часть — автоматизация не формальных выкладок, а того, что называют интуицией. Человек обычно не штурмует проблему в лоб, он не перебирает всех путей для ее решения, а выбирает только те, которые вследствие каких-то аналогий и не вполне познанных законов деятельности мозга кажутся ему ведущими к цели. Поэтому человеческий мозг добивается лучших результатов, чем машина, обладающая большой скоростью работы, но в которой не запрограммирована эта интуиция, потому что мы пока не знаем, что это такое.

В связи со сказанным, возникает новый, не менее важный вопрос: а зачем все это закладывать в машину? Если человек обладает интуицией, а как вложить ее в машину, мы не знаем, то давайте предоставим эту интуицию человеку. Человеку — человеческое, а машине — машинное. Машине оставим перебор вариантов и оформление окончательного варианта в виде, удобном для печати. Такая задача уже поставлена.

Но здесь мы подходим к одной из актуальнейших проблем развития всей электронно-вычислительной техники — проблеме общения человека

с машиной. Дело в том, что в электронной вычислительной машине сегодняшнего дня «кирпичики» ее элементарных действий очень сложны и нужен искусный каменщик для того, чтобы из этих кирпичиков сложить стройное здание математической программы. И если исследователю, который пользуется электронной вычислительной машиной для проверки доказательств, надо будет каждый раз «вкладывать» в нее интуицию, то он может сказать: «Бог с ней, с этой машиной, я сам скорее этот вариант решу вручную».

Проблема общения человека с машиной давно волнует коллектив Института кибернетики АН УССР, и мы пытались решить такого рода проблему для формульного вычисления, пытались сделать так, чтобы с машиной мог работать не только математик-программист, но и инженер. Для таких вычислений была создана электронная вычислительная машина «Промінь» малой производительности с упрощенным вводом и программированием. Она очень маленькая, размером с туалетный столик, но вследствие того, что тщательно продуман набор операций, команд, сделаны различные упрощения, становится возможным легко программировать простые задачи, с которыми часто сталкиваются конструкторы машин. И инженер через полчаса после того, как ознакомился с инструкцией, уже работает на этой машине, чувствует себя ее хозяином и она его «понимает». Поэтому инженеры с большой охотой идут на применение таких машин. Сейчас коллективом института создана новая, гораздо более совершенная машина аналогичного класса под названием «Мир» (машина для инженерных расчетов).

Однако все сказанное касается пока лишь формульных вычислений. Разработка такого языка, который был бы «понятен» машине и достаточно близок к естественному человеческому языку, потребует еще больших исследований. Пока создаются только первые наметки такого языка, и те программы, которые мы строим для автоматизации научного творчества, будут уже закодированы в «переводе» на этот язык. Это еще далеко не то, чего нам хотелось бы. Такое направление будет одним из основных в будущем развитии электронно-вычислительной техники.

Как видно, нам приходится не раз возвращаться к проблеме языка: «язык для экономических расчетов», «язык для описания схем электронных вычислительных машин», «точный язык» для общения с машиной — язык, язык и еще раз язык. Нередко и мы сами, специалисты по электронным вычислительным машинам, недооцениваем те сдвиги, которые происходят в науке (и прежде всего в математике) в связи с рождением подобных языков. По-видимому, нынешний этап развития математики можно сравнить с состоянием науки в XVII в., когда складывался формульный язык алгебры и математического анализа.

Что происходило тогда? Существовали описательные приемы решения тех или иных задач. Математики достигли большого искусства в использовании этих приемов. Когда были сделаны первые попытки формализации языка математики, создания языка дифференциального и интегрального исчислений, многие ученые говорили: «А зачем эти ухищрения, когда я старыми методами могу эту задачу решить? Вы утверждаете, что они позволяют провести касательную к этой кривой? А я это и так могу сделать».

Значение формульного языка не сразу осознается. И сам язык в науке создается постепенно, и значение его рождения становится ясным гораздо позже.

Мы можем сказать, что если бы подобный язык не был создан (в этом отношении характерна судьба японской математики), то в XVIII—XIX вв.

алгебра и анализ не получили бы такого развития. В принципе можно был все задачи решить старыми методами, но эти решения были бы лишен изящества, логичности, пришедшими в математику вместе с новыми мето дами, новым языком.

Но уже в XIX в. оказалось, что если говорить о средствах, выражаю щих конечный итог решения математиками какой-то задачи, то язык фор мул для этого недостаточен. Было время, когда математики верили, чт каждое дифференциальное уравнение можно решить в квадратурах интегралах, с помощью языка символов. Но было доказано, что существую уравнения, которые принципиально нельзя решить таким образом, был доказана неполнота этого языка.

Сейчас в науке рождается новый язык — алгоритмический, котором не свойственна эта ограниченность старого языка. Тем не менее многи математики относятся к этому языку примерно так же, как относились математики XVII в. к формульной символике.

Например, можно решить задачу и вывести формулу, написат интеграл — это хорошее, изящное решение. Можно вместо этого написат стандартную программу для электронной вычислительной машины, кото рая тоже решает проблему и с не меньшей степенью общности, че формула. Однако говорят, что это последнее решение численное, а то первое,— решение в общем виде. Но, в сущности, почему интеграл — ре шение в общем виде? Просто формульный язык символов нам понятен программу же мы часто не понимаем, она написана на незнакомом на языке.

Есть основание думать, что когда станет возможной краткая запис стандартной алгоритмической программы, скажем будет разработан алгебра формальных преобразований внутри машинного языка, то тако язык станет для математической логики столь же ясным и доступным как для нас язык формул. А формулы будут использоваться лишь в неслож ных случаях. Тогда математики привыкнут и к этим непривычным для на алгоритмам в записях и будут пользоваться ими для решения пробле которые принципиально нельзя решить в одной формульной записи.

Таким образом, проблемы вычислительной техники привели нас некоторым мыслям по поводу будущего всей науки. Электронные вычисл тельные машины имеют огромное значение для ее развития, и мы еще н представляем себе всех тех последствий, которые повлечет за собой испол вование средств автоматизации умственной деятельности человека.

В связи с перспективой автоматизации умственной деятельности з рубежом очень часто высказывают опасение, что если все будут дела машины, то человеку нечего будет делать, машина вытеснит его. Особен часто это приходится слышать, когда речь заходит об автоматизации нау ных исследований. Говорят, что ученым останется только почивать на ла рах и снимать урожай доказательств. Я вполне серьезно думаю, что чер 20—30 лет действительно можно будет наблюдать такие случаи. Скажем два ученых, один из которых более способный и более трудолюбивый, че другой, сидят рядом, причем первый не пользуется машиной для доказ тельств, а второй пользуется. И вот первый, более способный и более тр долюбивый, с удивлением видит, что он делает менее интересные вещ чем его сосед.

Но это вовсе не значит, что машина вытеснит человека. Просто задач стоящие перед человеком, неизмеримо усложнятся вместе с возможн стями их решения. В этом диалектика развития. Наверное, когда изобрел мотоцикл, тоже раздавались голоса, что, мол, теперь бегуны на длинну

дистанцию исчезнут и, значит, человечество «оскудеет» в физическом отношении. Но этого не произошло: рекорды, которые ставят сегодня наши спортсмены, и не снились спортсменам XIX в. Почему же мы должны бояться того, что, когда человек умножит свою интеллектуальную мощь с помощью этих машин, он будет ими вытеснен?

Когда была создана первая электронная вычислительная машина, кое-кто тоже опасался, что если машины работают с такой скоростью, то, когда мы создадим сотни таких машин, они за несколько минут перерешают все задачи. Однако появление машин вызвало к жизни еще большее количество задач, с которыми не только те машины, которые у нас есть, но и те, которые будут, не в состоянии справиться. И так как только человек может знать, что ему нужно, именно он будет ставить перед машинами задачи и направлять их на путь решения тех или иных проблем. Так что работы у человека хватит.

Конечно, вопрос о перспективах использования машин — это вопрос не только технический, но прежде всего социальный. Если в капиталистическом мире применение электронно-вычислительной техники в той или иной отрасли часто становится источником бедствий для трудящихся, например приводит к массовому увольнению банковских клерков, то в условиях социалистического строя внедрение электронных вычислительных машин будет способствовать общему повышению народного благосостояния. Мы должны смело смотреть в будущее и верить в то, что, какие бы удивительные применения ни нашла электронно-вычислительная техника в будущем, в условиях нашего общества она всегда будет служить на благо человека.

МОДЕЛИРОВАНИЕ МЫСЛИТЕЛЬНЫХ ПРОЦЕССОВ

(Природа.— 1963.— № 2)

Моделирование сложных мыслительных процессов — одна из самых увлекательных и вместе с тем самых сложных проблем кибернетики. Интерес к этой проблеме вызван двумя обстоятельствами. Во-первых, переход от простого наблюдения работы мыслительного аппарата человека к его активному моделированию позволит гораздо быстрее раскрыть многие тайны, и по сей день окутывающие процесс мышления. Во-вторых, моделирование мыслительных процессов с привлечением современной кибернетической техники служит основой для автоматизации многих видов умственной деятельности человека.

Модель мозга. В зависимости от целей, которые преследуют моделирование мыслительных процессов, оно может осуществляться разными путями. Различают два основных вида такого моделирования — прямое и косвенное или феноменологическое. При прямом моделировании основное внимание уделяется естественному мыслительному аппарату — мозгу человека. Моделирование собственно мыслительных процессов получается при этом как результат моделирования этого аппарата. При косвенном моделировании воспроизводится лишь общий ход течения мыслительного процесса закономерности перехода от одной мысли к другой. Способы же ревлизации (внутреннего механизма) таких переходов как правило имеют при этом мало общего с действительными процессами, протекающими в мозгу человека.

Биологов, изучающих мыслительный аппарат человека, разумеется, в первую очередь должен интересовать прямой метод моделирования. К сожалению, возможности этого метода в настоящее время весьма ограничены. Это связано с тем, что информационная модель нейрона, учитывающая многие известные тонкости его поведения, требует для своей реализации достаточно сложных радиоэлектронных схем. Огрубляя модель, удается снизить сложность соответствующих схем, однако при современном состоянии радиоэлектроники подобный процесс возможен лишь до известного предела. Для приблизительной ориентировки можно считать, что самая грубая модель нейрона имеет такой же порядок сложности, как одноламповый радиоприемник. Но при уточнении модели сложность этих схем более уместно сравнивать со сложностью современных многоламповых приемников и телевизоров. Разумеется, заменяя лампы полупроводниковыми или магнитными элементами, можно существенно уменьшить габариты схем, моделирующих нейтроны, однако их сложность (измеряемая количеством используемых в схеме деталей) при этом сохраняет прежний порядок. Относительно высокой остается и стоимость подобного рода моделей. А ведь человеческий мозг состоит не менее чем из десяти миллиардов нейронов! Если стоимость модели одного нейрона принять равной всего 10 коп., то лишь на моделирование всех составляющих его нейронов при-

пилось бы затратить миллиард рублей! А ведь в эту сумму не входят еще расходы на сборку и отладку схемы, которые, по-видимому, были бы еще более грандиозными! К тому же в настоящее время далеко не ясно, каким образом должны быть соединены между собой модели нейронов, чтобы образовать систему, действительно моделирующую мозг.

Таким образом, можно сказать, что при современном состоянии науки и техники задачи прямого моделирования человеческого мозга практически неосуществимы. Еще много нужно сделать в изучении строения мозга и в создании принципиально новых методов изготовления и монтажа радиоэлектронных элементов, чтобы осуществить моделирование этого сложнейшего и совершеннейшего живого органа. Пока же приходится ограничиваться гораздо более скромными целями, моделируя системы нейрона, состоящие из нескольких десятков или, в лучшем случае, нескольких сот нейронов.

Следует, однако, отметить, что уже такое, весьма скромное по своим масштабам моделирование может принести большую пользу биологам, изучающим мозг и происходящие в нем процессы на клеточном уровне. Используя модель системы нейронов, биолог может проверять гипотезы, касающиеся закономерностей передачи информации от нейрона к нейрону в различных участках мозга, более детально изучать механизмы возникновения и угасания условных рефлексов и т. д.

Серьезное препятствие на пути широкого использования биологами методов электронного моделирования нейронов и систем нейронов — относительно высокая стоимость соответствующего электронного оборудования и необходимость специального персонала для его эксплуатации, ремонта и наладки. Однако в настоящее время эти препятствия можно легко преодолеть, используя универсальные электронные цифровые машины, установленные в вычислительных центрах или в научно-исследовательских институтах. Дело в том, что функционирование любой модели (не только мозга, но и какого угодно другого объекта) можно имитировать при помощи универсальной цифровой машины, составляя и вводя в нее соответствующую программу. В эту программу входит цифровое описание модели и условий, в которых она находится, а машина описывает поведение этой модели. От экспериментатора требуется лишь умение составить программу. Искусство программирования вопреки мнению, бытующему среди далеких от математики людей, вовсе не является чем-то непостижимо сложным. Любой человек, имеющий среднее образование, может при желании за несколько месяцев научиться хорошо программировать.

Научившись же этому, экспериментатор получает возможность легко подготавливать и проводить на универсальных цифровых машинах эксперименты со столь сложными моделями, каких ему никогда не удалось бы создать своими руками. Расходы при таком подходе к моделированию сводятся, по сути, лишь к оплате машинного времени и не идут ни в какое сравнение с затратами на изготовление аппаратуры, соответствующей по функциям моделируемым органам.

К сожалению, биологи все еще крайне недостаточно используют новую мощную технику эксперимента, которая создана благодаря успехам современной кибернетики и вычислительной техники. В частности, немногочисленны пока попытки моделирования на универсальных цифровых машинах отдельных нейронов и систем нейронов, хотя уже первые, пока еще робкие эксперименты, показывают большие возможности, открывающиеся на этом пути. В качестве примеров успешного использования современных электронных вычислительных машин для моделирования эле-

ментов мыслительного аппарата человека можно отметить следующие факты: в США изучалось поведение системы из нескольких сот связанных между собой относительно грубых моделей нейронов. Недавно на одной из вычислительных машин Института кибернетики АН УССР было осуществлено моделирование хотя и одного нейрона, но с учетом многих тонкостей его поведения.

Несмотря на огромную принципиальную важность прямого моделирования мыслительного аппарата человека, отметим, что основой для реальной автоматизации мыслительных процессов в настоящее время могут служить не прямые, а косвенные методы. Это нетрудно понять, если вспомнить, что прямое моделирование, даже при условии использования электронных цифровых машин, может охватить пока элементы, состоящие из нескольких сот или в крайнем случае из нескольких тысяч нейронов. Во всяком же сколько-нибудь сложном виде умственной деятельности человека используется одновременно гораздо большее число нейронов головного мозга. Да и обязательно ли нужно при автоматизации сложных мыслительных процессов слепо следовать, естественному мыслительному аппарату — мозгу человека? История развития техники знает немало примеров, когда слепое копирование природы не только не двигало технику вперед, но часто и тормозило ее развитие. Достаточно вспомнить, что первые паровозы пытались снабдить «ногами», а на заре авиации много сил и энергии исследователей отняли попытки подражать маховым движениям крыльев птиц. Не следует забывать, что материальная основа сегодняшнего моделирования мыслительных процессов — электронные элементы — качественно отлична от материальной основы живых нейронов — живого белка. Механизм же моделирования должен отражать в первую очередь специфику его материальной основы. Те формы в организации взаимных связей и передачи информации, которые хороши для живых нейронов, вовсе не обязательно должны быть хорошими для моделирующих их электронных элементов, и наоборот.

Алгоритм и кодирование. На современном этапе развития кибернетики особое значение приобрели формы автоматизации различных участков умственной деятельности человека, основанные на их так называемом алгоритмическом описании. Понятие алгоритма, возникающее первоначально в математике, приобрело гораздо более универсальное значение. Алгоритмом называют любую конечную систему правил, позволяющих производить однозначное преобразование информации, заданной в обобщенной буквенной форме. Добавление термина «обобщенная» применительно к буквенной информации означает, что речь идет не обязательно о буквах, латинского или какого-нибудь иного применяемого для записи лексической информации алфавита. В общей теории алгоритмов рассматриваются обобщенные алфавиты, состоящие из любых индивидуально различимых знаков или символов. Существенно лишь, чтобы общее число различных знаков, составляющих алфавит, было конечным.

Вследствие столь широкого толкования понятия алфавита любой вид информации, с которым человек встречается на практике, можно представить в (обобщенной) буквенной форме. Процесс такого представления обычно называется кодированием. Термин «кодирование» (или «перекодирование») относят также к процессу перезаписи при помощи какого-либо алфавита информации, заданной первоначально в одном определенном алфавите.

При кодировании, как правило, стремятся пользоваться алфавитами с относительно небольшим числом букв. Например, поскольку общее

число различных позиций на шахматной доске конечно, можно было бы кодировать каждую позицию специально отведенным для нее символом — обобщенной буквой. Однако алфавит при этом был бы чудовищно велик. Поэтому на практике, как известно, предпочитают кодировать шахматные позиции при помощи алфавита, состоящего из части букв латинского и русского алфавитов и восьми цифр (от 1 до 8).

Еще один пример. Составляя рисунки заданного формата из черных и белых квадратиков фиксированного размера, можно получить конечное (хотя обычно и очень большое) число различных рисунков. Однако кодирование рисунков, использующее отдельную букву для каждого рисунка, практически неосуществимо (если только буквами не считать сами рисунки!). Поэтому более целесообразно ввести лишь две различные буквы для обозначения черного и белого квадратиков и кодировать рисунки последовательностями, составленными из этих букв (необходимо, разумеется, предварительно фиксировать порядок «обегания» рисунка, например по строкам или столбцам).

При кодировании информации в обобщенных алфавитах возникают обобщенные слова, т. е. конечные последовательности обобщенных букв. Поскольку знак раздела между словами может быть включен в обобщенный алфавит, то любую конечную последовательность обобщенных слов можно считать одним словом. Так обычно и поступают в общей теории алгоритмов. Для целей же более естественного моделирования мыслительных процессов целесообразно представлять себе информацию записанной в виде совокупности отдельных слов.

Имея в виду общность понятия алфавита, фактически любой вид умственной деятельности человека можно представить в виде преобразования буквенной информации. Если при этом к одной и той же входной информации человек всегда относит одну и ту же выходную информацию, а процесс преобразования входной информации в выходную описывается конечным (хотя, быть может, и очень большим) числом правил, то мы имеем алгоритм.

Пусть, например, мы имеем дело с шахматной игрой: входная информация представляет собой позицию перед очередным ходом, а выходная — после выполнения этого хода. Легко понять, что одни лишь правила шахматной игры сами по себе не составляют алгоритма, поскольку они не определяют, вообще однозначным образом очередной ход. Однако если дополнить эти правила специальными правилами, позволяющими оценивать различные позиции и выбирать каждый раз наилучший (с точки зрения данной системы правил) ход, то мы получим некоторый алгоритм игры в шахматы. Ясно, что существует не один, а много различных шахматных алгоритмов. Относительно нетрудно строить алгоритмы, моделирующие игру шахматистов низкой и даже средней квалификации. Гораздо труднее описать правила для моделирования игры сильных шахматистов. Эта проблема привлекает в настоящее время внимание целого ряда математиков (особенно в США) и, по-видимому, будет решена в ближайшие годы.

Что же дает алгоритмизация того или иного вида умственной деятельности? Оказывается, на современном уровне развития кибернетики алгоритмическое описание мыслительного процесса дает, как правило, возможность моделировать и автоматизировать его на базе уже существующих универсальных электронных цифровых машин.

Дело в том, что любую буквенную информацию нетрудно закодировать цифрами (для этого достаточно, например, заменить каждую букву ее порядковым номером в алфавите). После этого любой алгоритм можно запрограммировать, т. е. записать в виде некоторой последовательности

команд, выполняемых электронной цифровой машиной. Введя указанную последовательность (так называемую программу) в машину, мы заставляем ее реализовать исходный алгоритм.

Если первый этап автоматизации — алгоритмическое описание процесса — выполнен достаточно тщательно, то второй этап — программирование — представляет собою уже чисто техническую работу, хотя, быть может, и весьма громоздкую. Критерием, позволяющим судить о тщательности выполнения алгоритмического описания, может служить следующий контрольный опыт: если человек, совершенно незнакомый с алгоритмируемым процессом (например, не умеющий играть в шахматы), без всякой подготовки, руководствуясь лишь описанной системой правил (пусть очень медленно), может выполнять этот процесс, то этап алгоритмирования может считаться выполненным удовлетворительно, а соответствующая система правил — подготовленной к программированию. Природа же указанных правил может быть любой: наравне с математическими формулами подходят правила, сформулированные подобно правилам грамматики или правилам уличного движения. Важно лишь, чтобы при пользовании этими правилами не возникало никаких двусмысленностей или неясностей (чего, например, о современных правилах уличного движения сказать нельзя).

Возможна ли автоматизация научного творчества? Эффект автоматизации мыслительных процессов определяется прежде всего огромной скоростью и точностью работы современных электронных цифровых машин. Именно вследствие преимущества в скорости машина оказывается способной выполнять соответствующую работу лучше, чем человек, составивший для нее программу. Например, при игре в шахматы из-за того, что машина способна просматривать в единицу времени гораздо большее число вариантов, чем человек, она может регулярно обыгрывать составителя введенной в нее шахматной программы.

Возникающий подобным образом эффект кажущегося интеллектуального превосходства машины над человеком дает возможность не просто автоматизировать ту или иную сферу умственной деятельности человека, но и резко поднять производительность труда в этой сфере. Необходимость же в подобном росте производительности труда ощущается сегодня в целом ряде областей умственной деятельности.

Общеизвестными примерами сказанного являются научные и инженерные расчеты, техническое проектирование, планирование народного хозяйства, оптимальное управление производственными процессами, диспетчерская и информационная служба. В данной статье мы остановимся на ином примере, а именно на проблеме автоматизации научного творчества.

Речь идет не об автоматизации вспомогательных работ, сопутствующих почти каждому научному исследованию, например выполнении трудоемких расчетов (это успешно делается уже сегодня), поиску и реферированию необходимой литературы (методы автоматизации этой работы успешно разрабатываются и найдут применение в ближайшем будущем). Нас интересует сам процесс научного творчества и в первую очередь — в области точных наук (математики, физики, и т. д.).

В качестве примера рассмотрим математику. Процесс научного творчества здесь многогранен. Он включает в себя введение новых понятий, постановку новых проблем, доказательство теорем, построение примеров и контрпримеров и т. д. Из перечисленных задач выделим лишь одну, а именно доказательство (или опровержение) уже сформулированных теорем.

Существует широко распространенное мнение, что именно эта задача составляет основу научного творчества в области математики. Трудоемкость и сложность этой задачи в общем случае несомненна: если не считать времени, затрачиваемого на ознакомление с литературой, то поиски доказательства или опровержения теорем занимают львиную долю в бюджете времени каждого математика. Существуют примеры, когда на поиски доказательства одной теоремы затрачивались десятки лет упорного труда талантливых ученых. Ясно, что резкое увеличение производительности труда при доказательстве новых теорем (основанное на автоматизации) не только заметно ускорило бы темпы научного прогресса, но и дало бы возможность решать такие проблемы, которые «невооруженному» человеческому уму просто недоступны.

В настоящее время, когда автоматизация доказательств делает лишь первые робкие шаги, разумеется, преждевременно говорить о моделировании универсальных способностей доказывать теоремы во всех областях современной математики. Наиболее целесообразно на первых порах выделять относительно узкие области математики, составлять отдельные программы доказательств для каждой из них. Первые шаги в этом направлении уже сделаны. Так, известный математик Хао Ванг, работающий в настоящее время в США, разработал программу, при помощи которой универсальная электронная цифровая машина за несколько минут доказала около четырехсот теорем из известного труда по математической логике Principia Mathematica. Эта программа открыла путь к доказательствам и новых, никем ранее не доказанных теорем математической логики. В Институте кибернетики АН УССР разработана программа для доказательства или опровержения произвольных теорем относительно корней вещественных полиномов (на основе так называемого алгоритма Тарского).

Положение, однако, осложняется тем, что далеко не для всех областей современной математики оказывается возможным построить универсальные алгоритмы доказательства или опровержения всех теорем, какие только можно сформулировать в рамках данных областей. Отсутствие подобного, так называемого универсального разрешающего алгоритма доказано, например, для арифметики натурального ряда чисел. Вследствие знаменитой теоремы Гёделя в арифметике натуральных чисел можно сформулировать такие теоремы, которые нельзя ни доказать, ни опровергнуть!

Этот результат на первый взгляд кажется весьма мало обнадеживающим с точки зрения перспектив автоматизации доказательства теорем. Однако в действительности дело обстоит вовсе не так плохо. Ведь и человек может «запрограммировать» у себя в мозгу (в результате процесса обучения) бесконечное число методов, необходимых для установления истинности или ложности всех теорем в неразрешимых теориях (т. е. в таких теориях, в которых отсутствует универсальный разрешающий алгоритм). В результате даже самый изощренный математик при поисках доказательства теорем в неразрешимых теориях пользуется фактически не универсальными, а частными разрешающими алгоритмами. Хотя эти алгоритмы и не способны дать ответы на все вопросы в рамках рассматриваемой теории, они тем не менее на практике обычно приводят к хорошим результатам. Программируя эти алгоритмы, мы решаем задачи автоматизации доказательств в неразрешимых теориях, если не в принципиальном, то, во всяком случае, в практическом аспекте.

Подчеркнем, что механизм получения логических следствий из известных результатов, являющийся основой автоматизации доказательств,

хорошо известен и может быть относительно просто запрограммирован. Однако само по себе это не может обеспечить действенную автоматизацию, так как приводит, как правило, к столь большому перебору различных возможных вариантов поисков доказательства, что такой перебор оказывается недоступным даже для современных электронных цифровых машин, выполняющих сотни тысяч операций в секунду. Главная задача при построении доказывающих алгоритмов состоит поэтому в уменьшении перебора на основе более глубокого проникновения в методы математического мышления и особенно той его части, которую принято называть математической интуицией.

Здесь напрашивается аналогия с шахматной игрой. Набор фактов, с которыми математик имеет дело на каждом этапе доказательства теоремы, аналогичен шахматной позиции, а правила логики, при помощи которых выводятся следствия из известных фактов, аналогичны правилам, по которым ходят шахматные фигуры. С учетом лишь последних правил практически невозможно прийти к выигрышу в партии, насчитывающей несколько десятков ходов, точно так же при помощи одних лишь правил логики практически невозможно доказать сколько-нибудь сложную теорему. В обоих случаях препятствием на пути к успеху будет необходимость рассматривать слишком большое число вариантов. Поэтому приходится ставить промежуточные цели, достижение которых возможно за меньшее число ходов (шахматных или логических соответственно).

В шахматах постановка подобных целей определяется стратегическими правилами, основанными на оценке позиций. Что же касается теории доказательств, то в отличие от шахмат соответствующие правила здесь не сформулированы даже в первом приближении.

Потенциальные возможности автоматизации существуют и в других областях научного творчества, в частности, в постановке новых проблем и в построении новых теорий, обобщающих совокупность фактов. Однако в этом направлении сделано пока еще столь мало, что соответствующие проблемы можно считать лишь поставленными.

Несмотря на всю важность чисто алгоритмического подхода к проблеме автоматизации научного творчества, нельзя не отметить известной его узости. При таком подходе исключается возможность проявления какой-либо случайности и такое важное свойство мозга, как способность совершенствовать свои ответы в процессе работы, по мере накопления опыта. Можно, однако, так расширить понятие алгоритма, что оно будет включать в себя как алгоритм со случайными переходами, так и самосовершенствующиеся системы алгоритмов. Такие алгоритмы в широком смысле слова могут так же успешно программироваться и выполняться универсальными электронными цифровыми машинами, как и рассматривавшиеся ранее алгоритмы в узком смысле этого слова.

Моделирование процессов распознавания образов. Отметим одно принципиальное отличие самосовершенствующихся систем автоматизации доказательств от систем, использующих жесткие, неизменные алгоритмы. Если такие системы рассматривать как изолированные, не взаимодействующие с внешней средой, то возможность самосовершенствования не внесет ничего принципиально нового. Это же относится к случаю, когда воздействия внешней среды на систему можно рассматривать как результат работы некоторого алгоритма. В случае же «неалгоритмичности» внешней среды положение существенно меняется. Для самосовершенствующейся системы алгоритмов, взаимодействующих с подобной средой, может оказаться возможным, например, доказательство (за бесконечное время)

всех истинных теорем какой-либо неразрешимой теории, что заведомо невозможно для любой жесткой системы алгоритмов.

Практически необходимость перехода к самосовершенствующимся системам возникает обычно в случае, когда трудно найти жесткий алгоритм, решающий заданную проблему, а алгоритм нахождения требуемого алгоритма является хотя и весьма трудоемким, но простым с точки зрения его описания. В таком случае программируется не сам рабочий алгоритм, а алгоритм поиска или уточнения рабочего алгоритма. Задача же фактического нахождения рабочего алгоритма (и последующей работы по нему) поручается при этом машине.

Классический пример подобной ситуации — это задача распознавания зрительных образов. Если попытаться найти алгоритм (систему правил), позволяющий отличать женские лица от мужских, то нетрудно убедиться, что эта простая на первый взгляд задача оказывается в действительности очень сложной. Между тем человеческий мозг относительно легко справляется с этой задачей, но, разумеется, подобная легкость появляется лишь в результате более или менее длительного процесса приспособления зрительного центра, происходящего в раннем возрасте.

Естественно попытаться моделировать указанный процесс приспособления, основываясь на тех или иных гипотезах относительно механизма подобного приспособления. К настоящему времени предложено много моделей такого механизма. Все эти механизмы в более или менее явной форме используют идею запоминания одного или нескольких изображений каждого класса в качестве эталонов для всех классов изображений. Новые изображения сравниваются с эталонами и в зависимости от их близости к тем или иным эталонам относятся к соответствующему классу.

В Институте кибернетики АН УССР разработана и построена специальная приставка к универсальной электронной цифровой машине — так называемый универсальный читающий автомат. Эта приставка позволяет вводить в машину произвольные рисунки с учетом не только черных и белых полей, но и различных полутонов. Вследствие наличия универсальной цифровой машины становится возможным быстрое моделирование и опробывание различных способов распознавания образцов, включая системы с самосовершенствованием и самообучением. В настоящее время разработан и испытан целый ряд таких систем. Некоторые из них, например система, обучающаяся распознаванию геометрических фигур, довольно хорошо имитируют приспособительные функции мозга человека в части такого вида деятельности.

В связи с проблемой обучения распознаванию зрительных образцов заслуживает внимания специальная алгоритмическая схема для решения указанной проблемы, которая была предложена американским ученым Ф. Розенблаттом. Эта система, названная перцептроном, включает в себя, помимо чувствительных элементов (сетчатки), которые воспринимают изображение, некоторое множество довольно грубых моделей нейронов, связанных с сетчаткой случайным образом. Одна часть входных каналов нейронов является возбуждающей, а другая — тормозящей. Когда суммарное возбуждение нейрона превосходит его суммарное торможение на некоторую пороговую величину, нейрон возбуждается и передает некоторую величину, называемую весом нейрона, на специальное устройство, суммирующее веса всех подключенных к нему возбужденных нейронов. Таких сумматоров несколько, и нейроны делятся на несколько групп в соответствии с тем, к какому сумматору они подключены.

Показание каждого сумматора интерпретируется как «степень похо-

жести» показываемого перцептрону изображения на образ (класс изображений), заранее поставленный в соответствие данному сумматору. Специально введенный в схему перцептрона механизм поощрения увеличивает вес тех нейронов, которые возбуждаются изображением, относящимся к тому же образу, что и соответствующий этим нейронам сумматор. В противном же случае вес нейронов уменьшается. В результате моделируется некоторый приспособительный процесс: веса полезно работающих нейронов увеличиваются, а веса нейронов, которые не помогают или даже мешают правильной работе перцептрона, уменьшаются. Продолжая подобный процесс обучения, можно добиться, чтобы перцептрон научился правильно классифицировать изображения по заранее определенным группам (образам).

Возможен и несколько видоизмененный режим работы перцептрона (называемый режимом самообучения), при котором группировка изображений, к которым нужно стремиться, не задается заранее, а определяется перцептроном в процессе работы.

Приведенное автором теоретическое исследование работы перцептрона показало, что примененная в нем организация процесса обучения и особенно процесса самообучения далеко не всегда приводит к хорошим результатам. Многие важные черты, свойственные обучению и самообучению человека (применительно к задачам распознавания образцов), в схеме перцептрона отражены плохо или совсем не отражены. В настоящее время существует возможность гораздо более рационального моделирования тех процессов, на которые был рассчитан перцептрон. Некоторые из этих возможностей уже нашли свое воплощение в реальных программах, разработанных и опробованных в Институте кибернетики АН УССР и в ряде других институтов.

В целом работы по моделированию процессов распознавания образов развиваются достаточно быстрыми темпами. В первую очередь это касается зрительных образов и в меньшей степени — речевых сигналов. А ведь работа по распознаванию такого рода образов составляет значительную долю (хотя и выполняемую обычно бессознательно) работы человеческого мозга, что видно хотя бы из сравнения объемов зрительного и слухового центров с другими участками мозга.

Опыты обучения машин. В последнее время идеи самоорганизации и самосовершенствования начинают вторгаться и в процессы моделирования таких видов умственной деятельности, как логическое мышление, обучение языку и т. п. Особый интерес представляют семантические построения, обеспечивающие автоматизацию распознания смысла и обучения такому распознаванию. Опыты такого рода были успешно проделаны в Институте кибернетики АН УССР.

Первоначальная идея автора, положенная в основу автоматизации процесса обучения распознаванию смысла фраз, состояла в следующем. Предположим, что рассматриваются лишь такие фразы, которые имеют простейшую грамматическую конструкцию, а именно подлежащее — сказуемое. Пусть, далее, дан какой-либо набор существительных и глаголов, из которых будут составляться подобные фразы. Первоначально некоторый набор фраз указанного типа составляется человеком (учителем), причем при их составлении ограничиваются лишь такими фразами, которые, по мнению учителя, имеют смысл. Эти фразы одна за другой вводятся в машину и запоминаются ею.

Процесс запоминания организован так, что вначале машина осуществляет «голую зубрежку» осмысленных фраз, т. е. запоминает их без

всякого изменения. Однако при известных условиях характер запоминания изменяется. Это происходит тогда, когда число фраз с одним и тем же сказуемым превосходит некоторую заранее фиксируемую величину — так называемый коэффициент терпения.

Пусть, например, коэффициент терпения равен двум, а машине после двух осмысленных фраз: «профессор думает» и «студент думает» была сообщена новая осмысленная фраза: «мальчик думает». В таком случае машина, вместо того чтобы зазубрить эту новую фразу, вводит новое понятие для обозначения класса всех думающих и запоминает, что профессор, студент и мальчик относятся к этому классу.

Подобная перестройка характера запоминания пока не приводит к появлению у машины каких-либо новых сведений о множестве всех осмысленных фраз по сравнению с тем, что сообщил ей учитель. Принципиально иной эффект достигается введением нового процесса, называемого процессом экстраполяции осмысленности. Этот процесс также управляется неким коэффициентом, называемым коэффициентом осторожности. Чтобы понять суть указанного процесса, предположим, что коэффициент осторожности равен двум, а машиной уже образован класс думающих. Если теперь машине сообщить, что два каких-либо представителя класса думающих, скажем профессор и мальчик, могут также говорить, то машина экстраполирует заключение, что все думающие являются вместе с тем и говорящими. В результате машина делает правильный вывод, что фраза «студент говорит» является осмысленной, хотя она и не содержалась в числе осмысленных фраз, на которых было проведено обучение машины.

Разумеется, в результате описанного процесса экстраполяции осмысленности машина может прийти и к неверным выводам. Если бы, скажем, в уже рассмотренном примере первоначальный класс был бы образован не по признаку сочетаемости с глаголом «думать», а по признаку сочетаемости «стоять», то в результате процесса экстраполяции машина пришла бы к неправильному заключению, что все стоящие являются вместе с тем и говорящими.

Для уменьшения числа подобных ошибок машина перед экстраполяцией составляет предварительно несколько фраз с глаголом «говорить», выбирая существительные (подлежащие) из класса стоящих случайным образом. Сообщив эти фразы учителю, она спрашивает, осмысленны ли они. И лишь получив на этот вопрос утвердительный ответ, машина производит экстраполяцию.

На основе описанной идеи в Институте кибернетики АН УССР была построена программа для вычислительной машины «Киев», при помощи которой были проведены опыты по обучению машины смыслу фраз не только простейшей, но и более сложной грамматической конструкции. Во время этих опытов машина сама создавала понятия «человек» «мебель» и др.

Интересно, что, изменяя значения некоторых параметров (аналогичных описанным выше коэффициентам терпения и осторожности), введенных в программу, удается моделировать самые различные параметры и типы обучения — от голой зубрежки до склонности к крайне поспешным выводам и заключениям. Впрочем, оба эти крайних случая приводят к замедлению процесса обучения (в первом случае из-за отсутствия экстраполяции, а во втором — из-за частых ошибок и вызываемой ими необходимости перестройки (классов). Существуют (находимые пока опытным путем) наилучшие значения указанных коэффициентов, при которых процесс обучения происходит в среднем наиболее быстро. Эти значения сильно зависят от первоначального набора слов, из которых строятся фразы.

Из-за небольшого объема памяти машины «Киев» опыты по обучению проводились с весьма бедным словарем, насчитывающим всего около 100 слов. При наличии существенно большей памяти дальнейшее развитие описанных принципов может привести к построению программ для обучения машины тому или иному человеческому языку с учетом не только его синтаксиса, но и семантики.

Для изучения возможностей самоорганизующихся систем большой интерес представляют опыты по моделированию биологической эволюции. Такие опыты были проделаны в Институте кибернетики АН УССР на универсальной электронной цифровой машине «Киев». Моделировался некоторый весьма просто устроенный «мир», в котором действовал «закон природы», управляющий перемещением «пищи» из одних участков этого «мира» в другие. Обитающие в описанном «мире» «живые существа» моделировались в виде программно-реализованных автоматов.

Каждый автомат снабжается двумя счетчиками — счетчиком «жизни» и счетчиком «голода». Первый счетчик отсчитывает число моментов времени, прошедших со времени «рождения» автомата, а второй — число моментов времени, отделяющих автомат от последнего момента, когда автомат находится на участке пространства, снабженном «пищей». При достижении счетчиком «жизни» или счетчиком «голода» некоторых определенных заранее значений автомат «умирал», т. е., попросту говоря, исключался из дальнейших рассмотрений.

Перемещения автомата в «мире» определялись его состояниями, а переход автомата из одного состояния в другое осуществлялся в зависимости от состояния ближайших к нему участков «мира» (т. е. от наличия или отсутствия в них «пищи» или других автоматов). Специальная матрица, управляющая подобными переходами, первоначально задается случайно, а впоследствие испытывает еще небольшие случайные изменения («мутации») в процессе «размножения» автоматов. Был выбран простейший вид размножения путем деления, причем одна из образовавшихся половинок полностью наследовала структуру исходного автомата, а со второй происходило то же самое, за исключением случайной мутации в матрице переходов. Момент деления определялся подходящим «возрастом» автомата (показанием его счетчиков «жизни»), относительно малым показанием счетчика «голода» и наличием свободных соседних ячеек для вновь образующихся в результате деления автоматов. В результате проведенных опытов имитировался процесс естественного отбора и приспособления автоматов к выбранному «закону природы». Этот процесс при желании можно интерпретировать так же, как процесс «познания» коллективом автоматов соответствующего «закона природы», так как в результате естественного отбора возникают автоматы, в структуре матрицы переходов которых все более и более отражается выбранный «закон природы».

Принципиально не видно никаких ограничений для того, чтобы с помощью подобных моделей, использующих крайне простую исходную структуру автоматов, но весьма сложный процесс развития «природы», можно было бы получить весьма высокие формы приспособления к указанному процессу (вплоть до возникновения и совершенствования моделей человеческого состояния и сознательной человеческой деятельности).

И все же пока сделаны лишь первые шаги в построении общей теории самоорганизующихся систем как основы моделирования мыслительных процессов. Несомненно, что успехи этого направления в будущем помогут объяснить многие загадки мышления и создадут прочную базу для автоматизации сложных мыслительных процессов.

ИСПОЛЬЗОВАНИЕ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА В ДЕЯТЕЛЬНОСТИ ИНЖЕНЕРОВ

(Кибернетика, М.: Наука.— 1986)

Практическая постановка задачи моделирования интеллектуальной деятельности человека связана с появлением и развитием компьютеров. Уже с момента возникновения компьютеры начали использоваться для автоматизации сложных инженерных расчетов. Например, на первом советском компьютере МЭСМ уже в 1951—1952 гг. были выполнены расчеты для будущей линии электропередач от проектировавшейся в то время гигантской гидроэлектростанции на Волге.

Сложность программирования и подготовки данных на первой стадии развития компьютеров ограничивали их применение, не позволяли осуществить комплексную автоматизацию даже расчетной части инженерных проектов, не говоря уже о других элементах сложного труда проектантов. Возникла задача упрощения общения человека с компьютером. Эта работа проводилась как в направлении совершенствования самих компьютеров — так называемого хардвэра, так и в совершенствовании их программного обеспечения — софтвера. В первом направлении — это совершенствование систем иерархической памяти, появление широкого набора терминального оборудования для организации диалога пользователей с компьютером (алфавитно-цифровые и графические дисплеи и др.) для ввода и вывода графической информации, для работы с компьютерами, расположенными на большом удалении, через каналы связи, и др. Важное значение имеет направление повышения исходного (не программированного) машинного интеллекта за счет встраивания в хардвэр развитых алгоритмических языков, облегчающих организацию эффективного диалога с пользователем. Эта линия, начатая советскими миникомпьютерами серии «МИР», прочно вошла в практику мирового электронного машиностроения.

Новые возможности широкого использования компьютеров в инженерном труде открыло развитие сетей компьютеров с удаленными терминалами. Тем самым открылась принципиальная возможность объединения вычислительно-информационных мощностей в национальных и межнациональных масштабах и использования их непосредственно с рабочих мест каждым инженерно-техническим работником.

В области софтвэра развитие операционных систем и современных систем автоматизации программирования резко упростило эффективное использование сложных вычислительных комплексов и сетей компьютеров для решения широкого круга задач, интересующих современного инженера. В настоящее время широкое применение получили так называемые пакеты прикладных программ, ориентированные на различные области применений. В пакетах с достаточным уровнем «интеллектуальности» используется язык описания данных и программ их обработки, максимально приближенный к тем языкам, которыми соответствующие группы

пользователей привыкли описывать свои задачи при обычных «ручных» методах их решения.

Например, пакет программ для решения задач геометрической оптики в качестве исходных данных использует понятные любому специалисту-оптику описания комбинаций с указанием их взаимного расположения, форм поверхностей и оптических свойств материалов. Пакеты программ, рассчитывающие те или иные характеристики заданной системы (величины аберраций, увеличение, угол зрения и др.), имеют привычные для специалиста-оптика названия и легко вызываются для работы в любой последовательности после загрузки пакета в память компьютера. В подобный «интеллектуальный» пакет обычно включаются также специальные программы, позволяющие быстро изменять те или иные элементы данных (например, расстояния между линзами, кривизну поверхностей и т. д.), отображать заданную систему и рассчитанные ее характеристики на экранах дисплеев и т. п. Если к тому же используется достаточно мощная вычислительная система или сеть компьютеров, то, как правило, удается организовать одновременное пользование пакетом со многих рабочих мест, снабженных соответствующими терминальными устройствами. В целях разгрузки центральной части системы от многих подготовительных операций та или иная часть этих операций может выполняться непосредственно на рабочих местах, снабжаемых для этой цели специальными так называемыми интеллектуальными терминалами (использующими относительно дешевые мини- или микрокомпьютеры).

Кроме рассмотренного примера узко специализированного пакета существуют пакеты общеинженерного применения. Простейший из них — формульный пакет — позволяет вводить в компьютер любые формулы (записываемые в обычно принятом «немашинном» виде) и, подставляя в них данные также в естественной форме (например, $x = 1{,}272$; $\alpha = 0{,}1 \times \times 10^6$), производит необходимые расчеты. Впрочем, современное развитие микроэлектроники привело к появлению дешевых карманных программируемых компьютеров, позволяющих эффективно автоматизировать процесс вычислений по формулам (пока относительно несложным). Карманные электронные компьютеры вытесняют такой привычный для каждого инженера старшего поколения инструмент, как логарифмическая линейка. Особое место в практике автоматизации проектно-конструкторских работ занимают так называемые графические пакеты. Развитые графические пакеты позволяют решать достаточно широкий круг задач с чертежами как плоских, так и объемных конструкций. Это прежде всего задачи быстрого поиска нужных элементов чертежа (линий поверхностей, объемов) по их наименованиям, автоматического снятия размеров и др. Сюда же относятся программы, позволяющие осуществлять любые преобразования чертежей и связанные с ними расчеты, которые изучаются всеми студентами инженерных специальностей в курсах технического черчения и начертательной геометрии. В качестве примеров можно указать задачи изменения масштабов, сдвигов, поворотов, нахождения различных проекций и сечений и т. п., а также расчетные задачи типа определения расстояний площадей и объемов.

Обязательными составными частями графических пакетов являются программы ввода и вывода графической информации. В качестве устройств ввода используются специальные планшеты или графические дисплеи со световым пером для рисования эскизов. Имеются также устройства для автоматического ввода чертежей и рисунков. В качестве устройств вывода кроме графических дисплеев используются одно- и двухкоординатные

графопостроители (плоттеры), позволяющие получать готовые чертежи с соблюдением всех принятых в техническом черчении стандартов. Помимо различия в толщине и характере вычерчиваемых линий (сплошные, пунктирные), некоторые плоттеры позволяют вычерчивать линии различных цветов. При этом интеллектуальные пакеты должны «понимать» слова, которыми инженер-проектант описывает особенности вывода тех или иных линий. В трехмерном графическом пакете, разработанном в Институте кибернетики АН УССР для компьютера БЭСМ-6 (система АСПРО), предусмотрена возможность «сборки» чертежа из тех или иных стандартных элементов (деталей или узлов конструкций), хранящихся в специальном машинном архиве. «Листая» страницы архива путем нажима на специальную клавишу, инженер-проектировщик видит изображенные на этих страницах элементы конструкций в одном из узлов графического дисплея. Находя подходящий элемент, он может переместить его световым пером или специальным подвижным маркером в нужное место чертежа, после чего происходит автоматическое совмещение масштабов и вставка элемента в чертеж.

В случае больших размеров архива подобный постраничный поиск будет занимать слишком много времени. Для уменьшения времени поиска в пакет вводится программа поиска деталей в архиве по их словесным описаниям. При использовании стандартизованных описаний (шифров деталей) задача подобного поиска решается относительно просто. Одной из важных задач, которую предстоит решить в рамках создания искусственного интеллекта, является обучение компьютера пониманию любых словесных описаний на естественных человеческих языках. Определенные шаги в этом направлении уже сделаны, для полного решения задачи предстоит преодолеть еще немало трудностей.

Само собой разумеется, что по желанию пользователя архив может расширяться и корректироваться. С этой целью в развитых графических пакетах предусматривается возможность автоматического преобразования эскизов, рисуемых пользователем на дисплее или планшете, в точные чертежи. Разумеется, для этого пользователь должен ввести в компьютер, кроме эскиза, дополнительную информацию о точных размерах, масштабе чертежа и т. п. Наличие такой возможности наряду со средствами автоматического считывания чертежей позволяет пользователю непрерывно пополнять архив новыми элементами. Программа пакета должна также предусматривать возможность исключения из архива устаревшей информации, перенумерации страниц, изменения системы обозначений и т. п.

Многие графические пакеты снабжаются дополнительными программами, позволяющими автоматически или в режиме диалога с конструктором размещать плоские элементы в заданной площади или объемные элементы в заданном объеме. При этом используются различные методы оптимизации, позволяющие получать лучшие инженерные решения. Обычно такие методы специализируют применительно к тем или иным классам задач. В современной практике автоматизированного проектирования широкое распространение получили такие задачи, как оптимизация процесса вырезки заготовок (для последующей штамповки) из плоского прямоугольного листа, размещение радиотехнических элементов и соединяющих их проводников на многослойных платах, оптимизация конфигурации трубопроводов на химических предприятиях и т. п. В подобных задачах наряду с точными методами математического программирования используются различные эвристические методы, заимствованные из опыта работы квалифицированных проектантов.

Развитые графические пакеты, будучи совмещенными с соответствующими проблемно-ориентированными расчетными пакетами, позволяют создавать комплексные системы автоматизации проектно-конструкторского диалога человек — компьютер. При этом за человеком остается наиболее творческая часть процесса — проектирование, определение общего замысла конструкции и наброска эскизов как ее общего вида, так и отдельных (нестандартных) узлов.

Вся же расчетная и оформительская часть работы, вплоть до выдачи полной технической документации (чертежей, спецификаций и др.) в соответствии с принятыми стандартами, выполняется компьютером. Как показывает опыт, при этом достигается ускорение процесса проектирования (в ряде случаев в десятки раз) с одновременным улучшением качества проектных решений. Так, созданная Институтом кибернетики АН УССР совместно с одним из проектных институтов города Киева автоматизированная система проектирования сборных железобетонных зданий ускорила процесс проектирования более чем в 20 раз, а стоимость проектирования сократилась в 6—7 раз. Заметно улучшились и качественные показатели. Подобные системы созданы и продолжают создаваться и в других отраслях народного хозяйства. Анализ работ по созданию автоматизированных систем проектирования в машиностроении и строительстве приводит к постановке перед разработчиками компьютеров и их программного обеспечения одной важной задачи. Прежде всего, как уже отмечалось выше, современные компьютеры плохо приспособлены к обработке графической информации. Поэтому необходимо, чтобы при разработке компьютеров новых поколений одновременно разрабатывался бы стандарт представления графической информации в разрабатываемых компьютерах и развитый пакет программ для обработки графической информации. Как показывает несложный анализ, многие трудности, которые испытывают программисты, создающие графические пакеты для уже выпускаемых промышленностью машин, могут быть существенно уменьшены при внесении соответствующих изменений в хардвэр. Далее, заказчики (проектные институты и КБ) предъявляют различные требования к объемам оборудования систем автоматизации проектирования и к конфигурациям этого оборудования. Поэтому (как в ряде других областей применения) для целей автоматизации проектно-конструкторских работ промышленность должна перейти к разработке программно-технических комплексов, ориентированных на классы применений. Как пример рассмотрим комплекс, ориентированный на автоматизацию проектно-конструкторских работ в машиностроении и строительстве. Такой комплекс должен набираться из отдельных модулей, в качестве которых могут использоваться процессоры различной производительности, возможно, специализированные блоки памяти разных типов и периферийное оборудование, включая как интеллектуальные, так и обычные удаленные терминалы и специальные рабочие места конструкторов.

Применительно к каждой такой конфигурации должны генерироваться развитый пакет программ трехмерной графики и эффективная операционная система, удовлетворяющая принятым стандартам документирования на выходе и входе. Расчетные пакеты, кроме обычного формульного пакета, должны содержать все геометрические расчеты (включая автоматический выбор формул для определения длин и объемов), а также все расчетные методы курса общеинженерной подготовки: теоретической механики, сопротивления материалов, теории машин и механизмов, деталей машин, электротехники и др.

В состав комплекса должна, разумеется, входить мощная система автоматизации программирования для подготовки специфических расчетных программ, используемых лишь для определенных видов конструкций (судов, самолетов и т. п.). Причем система автоматизации программирования должна обеспечивать автоматическое считывание данных, генерируемых его программами непосредственно с хранящихся в машинной памяти чертежей. В комплексе, естественно должны быть предусмотрены средства ведения архива стандартных элементов конструкций, о котором уже говорилось выше. Поскольку в математическое обеспечение подобного комплекса включены наиболее трудоемкие программы, то его можно очень быстро приспособить к любым специфическим задачам.

Разумеется, описанный комплекс не охватывает всех областей автоматизации проектно-конструкторских работ. Понятно, например, что для проектирования радиоэлектронных изделий (в том числе компьютеров) желательно несколько видоизменить состав его первичного математического обеспечения. Аналогичное положение наблюдается и для ряда других инженерных специальностей (химическая технология, мелиорация, электрические сети и др.). Вообще, до сих пор мы рассматривали прежде всего лишь чисто конструкторский аспект проектирования (да и то лишь некоторую его часть). Между тем не менее важным аспектом процесса проектирования является проектирование технологии.

В качестве примера рассмотрим технологическое проектирование в машиностроении, скажем, проектирование технологии обработки деталей на металлорежущих станках. Здесь, как и на чисто конструкторском этапе, созданы пакеты программ, рассчитывающих и оптимизирующих процессы резания металла. Новым качеством, привнесенным в технологию металлообработки научно-технической революцией, является широкое распространение программно-управляемого оборудования и прежде всего станков с числовым программным управлением (ЧПУ).

Разработка технологии для станков ЧПУ является сегодня гораздо более трудоемким процессом, чем разработка обычной ручной технологии. Это связано с тем, что при современном уровне «интеллектуальности» ЧПУ выходной «документ» (лента для станка с ЧПУ) должен, как правило, определять каждое элементарное движение режущего инструмента и другие столь же мелкие подробности процесса, которые опытному рабочему не нужны. Поэтому появились специальные системы автоматизации подготовки программ для станков с ЧПУ. В этих системах используются особые языки, нацеленные на описание обрабатываемых поверхностей и режимов обработки, и производится автоматический перевод сделанных на этих языках описаний в окончательные технологические документы — ленты для станков с ЧПУ.

Развитие подобных систем выдвинуло два вопроса, требующих решения. Первый — проектирование системы автоматизированного конструкторского проектирования с системами автоматизации технологического проектирования. В результате такой интеграции для программно-управляемой технологии изготовления вновь спроектированного изделия можно устранить значительную часть обычной чертежной документации, оставив на нее только то, что необходимо человеку для контроля работы соответствующего программно-управляемого оборудования.

Второй вопрос — повышение уровня «интеллекта» программно-управляемого оборудования с целью упрощения приготовления необходимой для его работы информации. Для решения этого вопроса необходимо включать в состав программно-управляемого оборудования (станков или ста-

ночных линий с ЧПУ) мини- и микрокомпьютеры, способные в реальном темпе работы оборудования интерпретировать обобщенные команды на «технологических» языках достаточно высокого уровня в последовательности элементарных движений резца и других элементарных технологических операций.

Подобное оборудование для ряда технологических процессов уже создано. В качестве примера можно указать на разработанный в Институте кибернетики АН УССР специализированный миникомпьютер «Киев-70» для программного управления процессом электронно-лучевой обработки. Обобщенные технологические команды, которые интерпретирует «Киев-70», состоят из трех частей. Первая часть определяет вид геометрической фигуры, которая должна быть обработана электронным (или ионным) лучом (круг, сектор, прямоугольник, ряд точек и т. п.). Вторая часть команды задает размеры фигуры и ее расположение на обрабатываемой пластинке. Третья, собственно технологическая, часть команды определяет частоту, величину и число импульсов, которые должны быть посланы в каждую элементарную площадку (определяемую поперечным сечением луча) обрабатываемой площади.

Отметим, что за последние годы программно-управляемая технология получила широкое развитие в новых направлениях. Особо отметим появление и начало внедрения в практику универсальных промышленных роботов. От специализированного программно-управляемого оборудования универсальный робот отличается прежде всего наличием у универсального исполнительного органа-аналога человеческой руки. Будучи помещена в конвейер, такая «рука» (соответствующим образом запрограммированная) способна выполнять различные сборочные операции. Помещенная на программно-управляемую движущую тележку она способна работать как универсальное подъемно-транспортное устройство. Это устройство может перемещать детали от одного станка к другому, снимать и устанавливать эти детали на станок, менять режущий инструмент и т. п. Особо эффективным такое устройство становится тогда, когда оно снабжается аналогом человеческого глаза и встроенным в него дальномером.

Наличие подобных универсальных роботов наряду с программно-управляемыми станками, прессами и другим программно-управляемым оборудованием позволяет строить полностью автоматические цехи и целые машиностроительные заводы, способные быстро перестраиваться на выпуск различной продукции. Более того, не обладая энергией человеческих навыков, робот практически мгновенно может менять характер своей работы без потери производительности, иными словами, конвейер (и целый завод), на котором вместо людей работают роботы, способен в режиме массового, поточного производства выпускать индивидуализированную продукцию. Разумеется, использовать эту возможность следует разумно, так как в интересах конечного потребителя — человека — некоторые свойства изделий (например, внешний вид) желательно индивидуализировать, а другие (например, размер патрона в электролампе) оставить стандартными.

Как бы там ни было, а новые возможности, открывающиеся перед автоматизацией производственных процессов, выдвигают новые задачи и перед автоматизацией проектирования. Проект нового изделия, предназначенного для выпуска на полностью автоматизированном предприятии, не может заканчиваться обычными чертежами и описаниями. Выходная документация проекта — это полный набор согласованных друг с

другом программ работы всех единиц программно-управляемого оборудования (включая универсальные работы).

Необходимость точного согласования (во времени и пространстве) работы каждой единицы программно-управляемого оборудования в масштабах целого предприятия (а впоследствии и целых групп взаимосвязанных предприятий) приводит нас к другому классу задач, которые сегодня принято относить к организационному управлению. Таким образом, задача проектирования нового изделия должна оканчиваться не только проектированием технологии его изготовления (на данном заводе), но и проектированием организации всего процесса его изготовления (включая материально-техническое снабжение, организацию профилактики и ремонта оборудования и т. д.).

Разумеется, чтобы сделать решение такой задачи более простым и реальным, необходимо повышать уровень интеллектуальности всех единиц программно-управляемого оборудования. Особо следует подчеркнуть наличие достаточно интеллектуального координирующего центра — автоматического менеджера, способного эффективно интерпретировать обобщенные команды организационного управления, выдаваемые системой автоматизированного проектирования.

К слову сказать, задача проектирования организационного управления при организации выпуска новых изделий решается уже сегодня во многих АСУ на машиностроительных и приборостроительных заводах — это так называемая задача технической подготовки производства. Разумеется, сегодня эта задача решается применительно не к вполне автоматизированному производству. Да и сами организационные АСУ, предназначенные для управления не только механизмами, но и людьми, работают (и в этих условиях не могут работать иначе) в человеко-машинном, а не в чисто автоматическом режиме.

Широкое внедрение в народное хозяйство универсальных промышленных роботов сегодня уже не чисто научная, а скорее — экономическая задача, так как для управления такими роботами (особенно обладающими «зрением») используются весьма мощные и достаточно дорогие компьютеры. В период перехода от частичной к полной автоматизации производства будет происходить все большее и большее смещение деятельности инженеров-производственников от текущих проблем управления производством к перспективным проблемам его развития и совершенствования. Иными словами, повышение «интеллектуальности» средств автоматизации технологических и организационно-управленческих процессов будет приводить к непрерывному увеличению доли творческого труда в деятельности инженеров — производственников, к постепенному сближению характера их деятельности к деятельности инженеров — проектантов.

Уже сегодняшние, хорошо спроектированные АСУ организационного управления высвобождают инженеров-производственников от многих видов рутинного труда. Вместе с тем хорошие организационные АСУ представляют собой мощный инструмент для детального анализа технико-экономических показателей работы предприятий, выявления узких мест, мешающих развитию производства. Руководители, получившие в свои руки подобные АСУ, уже принимают меры для переориентации деятельности своих инженеров в этом направлении. Для этого необходимо, чтобы инженеры овладели новыми возможностями, которые дает им АСУ, активно использовали и развивали эти возможности, а освобождающееся от рутинных операций время тратили бы для творческого осмысливания проводимых с помощью АСУ анализов и выработки новатор-

ских творческих предложений по дальнейшему совершенствованию производства.

Для более полного раскрытия всех этих новых возможностей необходимо постоянное внимание к развитию измерительных и контрольно-испытательных функций АСУ. Необходимость ускоренного развития этих функций диктуется также постоянным усложнением современного производства, усложнением выпускаемых изделий и постоянным усилением требований к их качеству.

Успехи современной вычислительной техники привели сегодня к настоящей революции в технике измерений и контрольно-испытательных операций. Прежде всего появление дешевых микропроцессов (на базе электронных микросхем с большим уровнем интеграции) сделало возможным непосредственное их встраивание во многие измерительные приборы. Обладая свойствами запоминания и обработки измеряемой информации, измерительные приборы приобретают принципиально новые качества, освобождающие людей от многих рутинных операций. Например, лазерный дальномер со встроенным в него микропроцессором, будучи должным образом запрограммированным, вместо одного измерения в считанные секунды выполняет многие тысячи измерений, вычисляет среднее значение измеряемого расстояния, среднеквадратичную ошибку его измерения и выдает готовые результаты в цифровом виде.

Для организации надежного и высококачественного производства многих промышленных изделий уже на стадии проектирования технологии разрабатываются сложные системы автоматического контроля различных производственных операций и специальные системы тестов для испытаний на различных стадиях производства и отладки как самого изделия, так и отдельных его блоков и узлов. В состав программно-управляемого оборудования все в большей степени включаются программно-управляемые контрольно-испытательные стенды. Для испытаний особо сложных объектов создаются комплексные системы автоматизации испытаний. В таких системах комплексно решается целая группа вопросов. Во-первых, организация получения информации с большого количества (многих сотен, а иногда и тысяч) разнообразных датчиков, а также программного управления очередностью и частотой получения информации. Далее, в состав системы включается вычислительный комплекс, осуществляющий первичную и вторичную обработку поступающей информации и документальное оформление результатов такой обработки (графики, таблицы, текстовый материал).

Во многих случаях жесткие фиксированные заранее программы испытаний оказываются недостаточно эффективными, занимая слишком много времени и не выявляя с достаточной полнотой все скрытые дефекты. Поэтому в состав лучших систем автоматизации испытаний включаются средства оптимизации планирования и управления испытаниями, учитывающие результаты уже проведенных этапов испытаний.

В ряде случаев используется предварительная запись поступающей от датчиков информации (возможно, предварительно частично обработанной) на машинные носители (магнитные ленты, гибкие магнитные диски и др.) для последующей обработки в специальных информационно-вычислительных центрах. Таким образом строится большинство автоматизированных систем испытаний подвижных объектов (самолетов, судов и др.).

Автоматизация процессов испытаний и контроля в производстве тесно связана с автоматизацией управления технологическими процессами и часто является составной частью последних. Особенно ярко это прояв-

ляется в наладочных операциях, которые сегодня занимают все больший удельный вес в производстве сложных машин и приборов. Увеличивающаяся сложность изделий, с одной стороны, и широкая автоматизация процессов наладки — с другой, делают труд наладчиков все в большей и большей степени разновидностью труда инженеров. Здесь, как и в других случаях, автоматизация освобождает наладчиков в первую очередь от рутинных операций, постоянно увеличивая долю их творческого труда.

Аналогичное положение наблюдается для автоматизированных (человеко-машинных) систем управления сложными технологическими процессами самых различных классов. Операторы, работающие на таких системах, во многих случаях должны иметь инженерную подготовку с большой долей творческого, неформализуемого сегодня начала в характере их труда.

Важное значение для инженеров всех специальностей и направлений деятельности (особенно для проектантов, конструкторов и технологов) имеют различного рода автоматизированные справочно-информационные системы. Прежде всего речь идет об автоматизации поиска необходимой патентной и справочной информации.

Так, конструктора новых машин могут интересовать данные о новейших конструкционных материалах, инженера-химика — о методах синтеза новых химических соединений и т. п. Первоначально такие системы создавались локально, в масштабах отдельных отраслей народного хозяйства или даже отдельных фирм и институтов. Появление и развитие сетей компьютеров и техники работы с компьютерами с удаленных терминалов сделало актуальной задачу создания автоматизированных справочно-информационных систем (обслуживающих инженеров и научных работников различных специальностей) в национальных и даже в межнациональных масштабах. Определенные шаги в этом направлении в ряде стран уже сделаны. Однако немало вопросов еще ждут своего решения. Среди таких вопросов — организация эффективных, широко доступных (для инженеров и научных работников соответствующих специальностей) автоматизированных фондов информации, получаемой в результате дорогостоящих или уникальных исследований. Причем речь идет о фондах первичной информации, полученной при измерениях, а не о сделанных в ходе ее обработки выводах и результатах. Смысл создания таких фондов состоит в возможности последующей обработки этой информации с помощью новых методов или под углом зрения новых задач. В качестве примера такой информации можно отметить данные, получаемые в результате геофизической разведки новых месторождений полезных ископаемых.

В справочно-информационных системах, имеющих дело с новыми идеями (например, с патентной информацией), оказывается возможным и целесообразным увеличивать уровень их «интеллектуальности». Помимо большей свободы в отношении языка запросов, в последнее время появились возможности наряду с новыми идеями, зафиксированными в системе автоматически получать от нее различного рода идеи, возникающие в результате комбинаций уже известных идей. Такой подход (применительно к новым результатам в области дедуктивных наук) развивается в настоящее время в Институте кибернетики АН УССР. Кроме собственно справочно-информационной системы он включает в себя систему интерпретации специально разработанного для этой цели языка практической математической логики (представляющего собой расширенный фрагмент обычного русского языка) и так называемого алгоритма очевидности, автоматически выводящего очевидные (с точки зрения этого алгорит-

ма) новые факты, вытекающие логически из известных систем фактов. Здесь мы вплотную подходим к проблемам автоматизации таких творческих элементов процесса проектирования, которые до сих пор оставались прерогативой человека. Ведь хорошо известно, что многие удачные проектно-конструкторские решения (в том числе и обладающие патентной чистотой) возникают в результате счастливых комбинаций уже известных идей. Сегодня ряд ученых предпринимает попытки автоматизировать процесс возникновения новых конструкторских идей как результат комбинаций уже известных. Практически полезные системы подобного вида потребуют по всей видимости, организации диалога с человеком для более эффективного поиска удачных комбинаций за счет отбрасывания заведомо бесперспективных направлений такого поиска.

Еще одно перспективное направление автоматизации более высоких творческих разделов инженерно-конструкторского труда — разработка так называемых алгебр конструкций. Алгебра конструкций должна в первую очередь иметь специальный символьно-цифровой язык, позволяющий записывать конструктивные решения в виде аналогов (как правило, значительно более сложных) обычных алгебраических формул. Далее, каждой такой алгебре сопоставляется система эквивалентных преобразований ее формул, не меняющих некоторые, заранее фиксированные инварианты (определяющие функциональное назначение конструкции). В обычной алгебре, например, преобразование формулы $ab + ac$ в формулу $a(b + c)$, изменяя внешний вид формулы, не меняет вычисляемого по ней значения. Это значение и будет в данном случае инвариантом преобразования. В случае, если для того или иного класса конструкций удается построить алгебру, задача инженера-конструктора состоит в том, чтобы найти какое-то (не обязательно лучшее или даже просто хорошее) решение стоящей перед ним конструкторской проблемы. Представив это решение в виде алгебраической формулы, он может подвергнуть его формальным эквивалентным преобразованиям и получить в результате некоторого целенаправленного поиска лучшее (если не самое лучшее из всех возможных) решение. Для сложных алгебр такие преобразования лучше всего выполнять в диалоговом (человеко-машинном) режиме, поручая всю технику эквивалентных преобразований и расчеты эффективности получаемых решений компьютеру.

Особенно эффективным подобный метод является в том случае, когда удается построить полную схему эквивалентных преобразований, т. е. такую, что в результате преобразований из любого данного конструкторского решения можно получить любое иное решение с тем же функциональным назначением (системой инвариантов). Автору удалось построить такую алгебру для решения задач проектирования компьютеров. Алгебры с достаточно богатыми системами эквивалентных преобразований построены для задач проектирования некоторых классов электрических схем, шарнирно-рычажных механизмов и др. Однако в целом задача алгебраизации конструкторских решений еще далека от полного решения.

Еще одной важной областью применения компьютеров для решения инженерных задач является системный анализ и управление разработками сложных (больших) систем. Актуальность задач этого класса определяется тем, что сегодня все чаще приходится разрабатывать и реализовывать инженерные проекты столь большой сложности, что их невозможно в сколько-нибудь полном виде охватить одному человеку. Разбивая же задачу на части и поручая их решение различным специалистам, мы, как правило, утрачиваем представление о системе в целом, результатом

чего могут быть непредвиденные отрицательные (а иногда и катастрофические) последствия ее функционирования.

Выход из данного положения может дать предварительное моделирование поведения системы на компьютерах (как правило, особо мощных). Для облегчения подобного моделирования разрабатываются специальные языки, направленные не столько на описание конструктивных особенностей отдельных элементов системы, сколько на описание поведения таких элементов в их взаимодействии. Специальная система программ позволяет по этим описаниям восстановить шаг за шагом поведение рассматриваемой системы во времени и пространстве в различных внешних условиях (также описываемых на соответствующем языке). Строящиеся таким образом автоматизированные системы моделирования сложных систем обычно осуществляют также автоматическое документирование результатов моделирования. В лучших же системах в той или иной мере осуществляется автоматизация процесса планирования и управления машинными экспериментами, составляющими суть процесса моделирования.

Очень важной особенностью подобных систем является то, что отдельные их части могут описываться специалистами в самых различных областях науки и техники, часто даже плохо понимающих друг друга. В машинной же модели восстанавливается та необходимая для анализа ее поведения целостность восприятия системы, которая утрачена людьми вследствие исторически вынужденного процесса специализаций знания.

Отметим, что современные методы математического моделирования (с использованием компьютеров) применимы и там, где отсутствуют классические математические описания (в виде формул, уравнений и т. п.) и даже возможность точного количественного определения интересующих нас параметров. Это особенно ценно в связи с тем, что многие современные инженерные проекты связаны с экологическими и социальными проблемами, плохо и даже совсем не приспособленными для обычных числовых характеристик.

Подобные системы, представляющие своеобразный синтез машинного и коллективного человеческого интеллекта, успешно используются сегодня для анализа сложных систем социальной и биологической природы, в которых нет ни одного количественного параметра, характеризуемого обычными числами. Такого рода системы успешно используются для прогноза развития науки и техники и управления научно-техническим прогрессом. Одна из возможных методик создания подобного рода систем (в двух вариантах) была предложена автором и ныне реализована в действующих системах.

В заключение отметим, что разработка сложных инженерных проектов требует (даже с учетом развития автоматизации проектирования) согласованной работы многотысячных коллективов инженеров различных специальностей. Для эффективного планирования и управления такими разработками необходимы специальные автоматизированные (человеко-машинные) системы управления. В таких системах используется специальный математический аппарат, сложная техника, требуются специальные технологические и организационно-управленческие знания. Необходимость создания и работы с данными системами связана с потребностью еще одной инженерной специальности, а именно инженера-организатора управления разработками. Такое же положение наблюдается в других областях, где используются АСУ. Таким образом, широкое внедрение компьютеров не только меняет характер труда инженеров традиционных специальностей, но и создает новые инженерные специальности.

СОДЕРЖАНИЕ

НАУЧНОЕ ИЗДАНИЕ

Глушков Виктор Михайлович

КИБЕРНЕТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА, ИНФОРМАТИКА

Том 2

ЭВМ — ТЕХНИЧЕСКАЯ БАЗА КИБЕРНЕТИКИ

Оформление художника *В. Г. Самсонова*
Художественный редактор *И. П. Антонюк*
Технический редактор *Г. М. Ковалева*
Корректоры *Е. А. Михалец, В. И. Божок*

ИБ № 10857

Сдано в набор 30.10.89. Подп. в печ. 23.04.90. БФ 08057. Формат 70×100/16. Бум. тип. № 1. Обыкн. нов. гарн. Выс. печ. Физ. печ. л. 16,75+1 вкл. на мел. бум. Усл. печ. л. 21,94. Усл. кр.-отт. 21,94. Уч.-изд. л. 21,91. Тираж 2930 экз. Заказ № 9—3611. Цена 4 р. 50 к.

Издательство «Наукова думка». 252601 Киев 4, ул. Репина, 3.

Отпечатано с матриц Головного предприятия республиканского производственного объединения «Полиграфкнига». 252057, Киев, ул. Довженко, 3 в Киевской книжной типографии научной книги. 252004 Киев 4, ул. Репина, 4. Зак. 0-450.