кажутся не слишком перспективными, и сосредоточивает свое внимание на небольшой группе вариантов (а частенько и на одном единственном варианте), которые кажутся ему наиболее многообещающими. Этот процесс называют *эвристикой**), а приемы, лежащие в основе отсечения неперспективных вариантов,— *эвристиками.* Если в программе для ЭВМ используются какие-либо эвристики, то сама программа называется обычно *эвристической.*

В отличие от точной алгоритмической программы, которая реализует алгоритм в, следовательно, обладает результативностью, эвристическая программа не гарантирует получения решения. Подобно человеку, она может ошибаться, не учитывать каких-то важных факторов. Именно так построены шахматные программы. Если бы они были не эвристическими, а алгоритмическими, то ЭВМ никогда бы не проигрывали. Как в игре в «крестики-нолики», они Действовали бы оптимальным образом, приводя ЭВМ к победе или ничьей, а в играх, где ситуация сложилась не в пользу ЭВМ, «мужественно» боролись бы до конца, перехватывая инициативу при любой мыслимой ошибке противника.

Эвристические программы нашли широкое применение при имитации интеллектуальных процессов. Примером этому могут служить разнообразные программы для сочинения музыкальных произведений на ЭВМ.

...Этого человека знают все специалисты-кибернетики нашей страны. Известен он и за рубежом. В Италии он считается одним из ведущих специалистов в области машинного сочинения музыки и его принимают в члены специального общества поклонников машинного творчества. В США, Франции и ЧССР издаются его работы. С неизменной виолончелью выступает он перед многочисленными аудиториями, по радио и телевидению, демонстрируя успехи ЭВМ в этой области. Это Р. Х. Зарипов. Он, как и многие другие кибернетики, совмещает одновременно много профессий: он и математик, и программист, и музыкант, и психолог. Машинному сочинению музыки он посвятил всю свою жизнь. И добился весьма многого. Его программы сегодня, по-видимому, лучшие в мире. А машинная музыка уже

*) Понятие «эвристика», конечно, шире того, как это здесь толкуется. Но для нас такого толкования вполне достаточно.



находится на уровне тех многочисленных музыкальных поделок, которых, к сожалению, так много.

Как же работает программа, сочиняющая музыку? Поясним это, опуская многочисленные детали, на примере одной из программ, разработанных Р. Х. Зариновым. Эта программа предназначена для сочинения однотональных мелодий в миноре или мажоре для двух видов тактового размера: 3/4 или 4/4. Сочиняется восьмитактный период с двумя музыкальными предложениями по четыре такта в каждом из них с учетом повторности ритмических и мелодических фигур, а также ладогармонических функций *).

На основании законов сочинения музыки, восходящих еще к Палестрине, можно написать ряд законов композиции, носящих эвристический характер. Эти законы отражают особенности человеческого восприятия музыки. Например, если разница по высоте звука между двумя соседними потами превысит определенный порог, то человеческое ухо воспримет это место как «плохо звучащее, режущее слух». Если звуковысотный ряд движется плавно и монотонно, высоты звуков монотонно возрастают или убывают, сохраняя приемлемые для человека пороги изменения, то музыка ласкает его слух, приятна и вызывает хорошие ассоциации.

Пусть теперь с помощью некоторого случайного механизма ЭВМ порождает на каждом шаге сочинения мелодии какую-то ноту (в реальной программе этот шаг расчленен на несколько самостоятельных шагов: выбор высоты ноты, выбор длительности ноты, выбор ритмической нагрузки ноты и т. п.). Нота может быть принята или не принята из-за требований, выдвигаемых правилами композиции. Эти правила, как уже говорилось, отражают специфику человеческого восприятия музыки. Но, кроме того, они еще отражают те внутренние ограничения на структуру будущего сочинения, которые заложены в нее программистом. Эти ограничения касаются допустимых вариантов гармонической структуры, допустимых вариантов мелодической структуры и допустимых вариантов масштабной структуры. Вся совокупность правил композиции отсекает

*) Читатель, не знакомый с терминами из теории музыки, может не обращать на них внимания, ибо суть программы будет ему понятна из дальнейшего.

от случайных последовательностей те последовательности, которые неприемлемы.

Можно представить себе все возможные восьмитактные мелодии в виде «раскидистого» дерева, вершинам которого соответствуют уже сочиненные начальные части мелодий, а стрелкам — возможные выборы нот в данной позиции. Тогда правила композиции производят отсечение целых поддеревьев, которые начинаются от отбрасываемой ноты, как от комля. Таким образом, программа Р. Х. Зарипова действует по тому же принципу, что и шахматные программы. Только просмотр и отбор вариантов продолжения процесса здесь происходит случайным образом. Музыкальное произведение в программе Р. Х. Зарипова рождается на основе трех последовательных шагов-этапов. На нервом этапе порождается общая структура — схема мелодии. На втором этапе на эту схему как бы накладывается ритмический рисунок будущего произведения. И только на третьем этапе происходит интервально-высотное наполнение ритмической структуры. В результате этой процедуры родились «Уральские напевы», сочиненные на машине «Урал-2» по программе Р. Х. Зарипова.

Так работают эвристики в процессе сочинения музыкальных произведений. Так же они работают и в собственно игровых программах и в программах, имитирующих решение различных комбинаторных проблем, связанных с распределением ограниченных ресурсов, выбором той или иной структуры проектируемого устройства и во многих других случаях. Это свидетельствует об универсальности подхода, основанного на применении эвристик. И это действительно так. Как станет ясно из дальнейшего, процедуры ограничения перебора являются специальными метапроцедурами, присущими различным уровням интеллектуальной деятельности.

**Узнавание.** Проблема узнавания — одна из самых захватывающих. Уже на заре развития кибернетики и вычислительных машин предпринимались попытки построения процедур, с помощью которых можно было бы проводить классификацию различных объектов на основании выделенной группы признаков. Такие классификации лежат в основе всей жизнедеятельности живых организмов, в том числе и человека. Всем организмам необходимо отличать то, что может быть пищей.



от того, что вредно подействует на организм, если его съесть. Каждое достаточно развитое живое существо должно отличать опасные ситуации от неопасных. Человек настолько связан с различными классифицирующими системами, что даже не отдает себе отчета в их существовании. Посмотрите на рис. 6. На нем изображены шесть геометрических фигур. Попробуйте расклассифицировать их. Результаты этой классификации могут быть различными. Кто-то выделит класс прямоугольников, класс шестиугольников и класс окружностей. Другой разделит фигуры на большие и маленькие. Третий отделит окружности от многоугольников. Но важным является то, что хотя условия распределения объектов по классам никак не сформулированы заранее, все люди способны использовать какие-то признаки, кажущиеся им существенными для классификации. Однако выбор тех или иных признаков для разбиения объектов на классы далеко не случаен. Человек лишь в принципе может осуществить любую классификацию. Если его не ставить в специальные экспе-

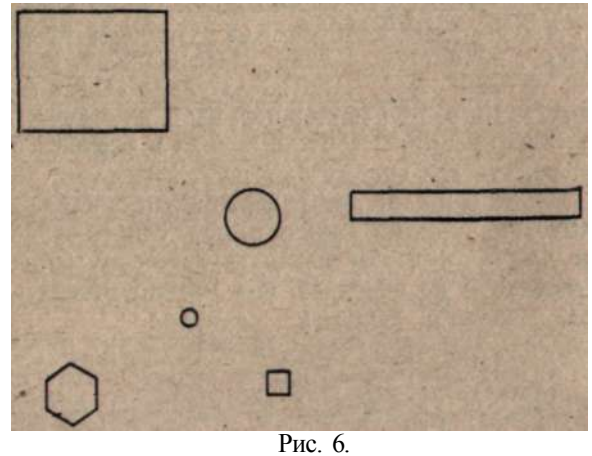


Рис. 6.

риментальные условия, то он проведет классификацию вполне определенного типа, а именно ту, которая подсказана ему жизненной практикой и в полезности которой он убедился на опыте.

При диагностике некоторых психических заболеваний с успехом используется тест на классификацию. Если вместо ожидаемого результата человек дает ре-

зультат, логически правильный, но неверный с точки зрения человеческой практики, то это усиливает «одобрение в его неблагополучии. Приведем классический пример использования такого теста. Испытуемому предъявляются четыре предмета: барабан, буденовка, шашка и зонтик. Задание состоит в отделении из этой группы лишнего предмета. Другими словами, перед испытуемым ставится задача классификации этих четырех объектов по двум классам при дополнительном условии, что один из этих классов должен содержать три предмета, а второй — один.

Ожидается, что люди должны разделить предметы таким образом, что зонтик попадает в один класс, а остальные предметы — в другой. Связано это с ситуативным использованием классифицируемых предметов. Трудно себе представить человека в буденовке с шашкой и зонтиком. Зонтик ситуативно не совместим с остальными предметами предъявленного набора. А именно ситуативная совместимость, постигнутая из жизненного опыта людей, является основой наиболее устойчивых классификационных систем у человека. Если же испытуемый отделяет от остальных предметов буденовку, объединяя в один класс барабан, шашку и зонтик, то экспериментатор обязательно задаст вопрос: «Почему?». На это он может получить, например, такой ответ: «Три предмета: зонтик, шашка и барабан имеют общим то, что все они издают звук, когда ими пользуются. Зонтик щелкает при раскрывании, шашка звенит при вытаскивании из ножен и при ударе, барабан и сделан для того, чтобы издавать звуки. А буденовка этим свойством не обладает, она мягкая. Когда ее надевают, никакого звука не слышно».

Правильно ли проведена классификация? Логически вполне правильно. Выбран классифицирующий признак и согласно ему произведено разбиение на два класса. Но тем не менее это решение нам как-то не по душе.

Значит, проблема классификации не может рассматриваться только на абстрактном уровне. ЭВМ легко могла бы найти многие способы классификации, но для нас интересно смоделировать процесс классификации в том виде, как это присуще человеку. А это требует специальной процедуры обучения классификации. Она особенно необходима в тех случаях (а, как показывает практика, их большинство), когда человек, умея классифицировать, не может точно сформулировать признаки, по которым эта классификация производится. На этом основано различие двух процедур: распознавания образов и узнавания. При распознавании образов, когда пространство признаков, в рамках которого можно построить классификацию, задано, проблема не столь сложна, как во втором случае.



Человек может сообщить машине лишь список конкретных примеров классификации, а ЭВМ на их основе сама должна построить классифицирующую систему, которую от нее ожидает человек. Конечно, такая постановка проблемы означает, что ЭВМ никогда не сможет получить точную классификацию. Ведь она строит ее на основе конкретных примеров, индуктивно. И лишь информация, содержащаяся в этих примерах, служит основой машинной классификации.

Тем не менее проблема узнавания с помощью построения классификации на основании примеров оказалась вполне реализуемой на ЭВМ. Несомненная заслуга в этом принадлежит еще одному замечательному советскому кибернетику Михаилу Моисеевичу Бонгарду. Подобно М. Л. Цетлину, М. М. Бонгард обладал той широтой взглядов, которая столь необходима при исследованиях в пограничных областях науки. Как и М. Л. Цетлин, М. М. Бонгард тесно сотрудничал с биологами, биофизиками, физиологами. Поэтому разработанные им процедуры, связанные с узнаванием, столь похожи на те, которыми пользуются люди.

Опишем суть одной из таких процедур, на простом примере. Пусть необходимо провести классификацию фигур, вырезанных из картона и покрашенных в различные цвета (рис. 7). Так как рисунок черно-белый, то цвета фигур обозначены написанными на них буквами: К —красный, З—зеленый и С—синий. Фигуры разложены в виде двух наборов, слева от вертикальной черты и справа от нее. Левые фигуры принадлежат одному классу, а правые — второму. Это так называемая *обучающая выборка.* Задача ЭВМ — определить принцип классификации, отраженный в обучающей выборке, и классифицировать любые другие фигуры в соответствии с найденной классификацией (узнавать класс, к которому они принадлежат).

Предположим, что программа узнавания, реализованная на ЭВМ, может получать информацию о форме

наблюдаемых предметов (круги или многоугольники), их цвете (красный, зеленый или синий) и размерах; (большой или маленький). Пусть программа способна формировать высказывания типа «Эта фигура большая» или «Эта фигура зеленая». К этим высказываниям мы можем применять операции конъюнкции и отрицания исчисления высказываний, о которых мы уже говорили ранее. Тогда станет возможным получать сложные высказывания вида «Эта фигура большая и зеленая» *), «Эта фигура маленькая и не красная». Относительно

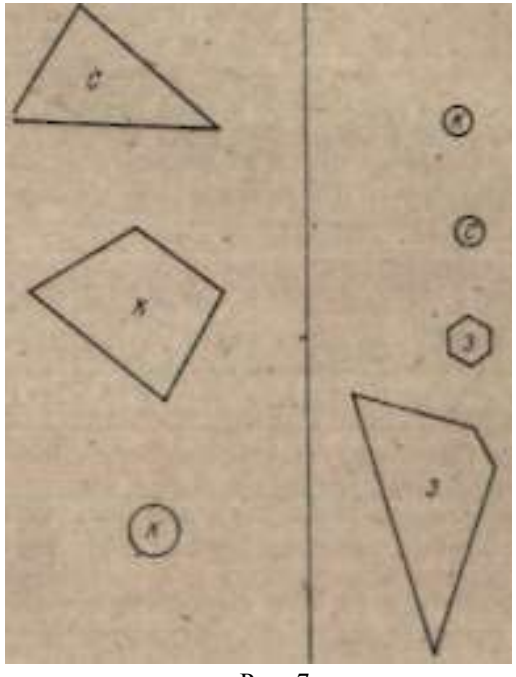


Рис. 7.

любой фигуры все эти высказывания могут быть как истинными, так и ложными. Например, высказывании «Эта фигура многоугольник и синяя» но отношению к первой фигуре левого набора на рисунке является истинным, но по отношению ко второй фигуре этого же левого набора это высказывание является ложным.

Признаком для совокупности фигур набора будем считать дизъюнкцию полученных для отдельных фигур

*) Для наглядности мы заменяем формулу и операции словами.



набора истинных высказывании. Например, для левого набора эта дизъюнкция может иметь вид: «"Эта фигура многоугольник и синяя" или „Эта фигура не круглая", или "Эта фигура красная"». Таких признаков для набора можно получить много. Примером другого признака для того же левого набора фигур может служить: «„Эта фигура не желтая" или „Эта фигура многоугольник", или „Эта фигура круглая и большая"».

Особенностью признака является то, что он принимает истинное значение для всех фигур данного набора. Отберем из них только те, которые на правом наборе не являются истинными для всех входящих в него фигур. Другими словами, возьмем только информативные признаки, с помощью которых можно отличить некоторые фигуры левого набора от некоторых фигур правого набора. Например, первый из построенных нами признаков для соответствующих фигур правого набора принимает следующие значения: истина, ложь, истина, истина. Поэтому он является информативным.

Образуем конъюнкцию из некоторого числа информативных признаков. Если эта конъюнкция обладает тем свойством, что для всех фигур левого набора она принимает значение «истина», а для всех фигур правого набора — значение «ложь», то ее называют *решающим правилом* или *правилом классификации.*

Для нашего примера такое правило после соответствующих преобразований высказываний может иметь, например, такой вид: «Эта фигура синий или красный многоугольник пли большой круг». Для всех фигур правого набора это высказывание ложно, а для всех фигур левого набора оно истинно.

Для правого набора решающее правило не обязано совпадать с отрицанием решающего правила для левого набора. Оно, например, может иметь вид «Эта фигура маленькая или зеленый многоугольник».

Верность найденного решающего правила связана с тем, как программа будет классифицировать другие объекты, не входившие в обучающую выборку. Ведь человек мог иметь в виду иную классифицирующую систему, чем найденная ЭВМ в процессе обучения. Например, человек мог относить к первому классу все большие фигуры кроме зеленых многоугольников, а ко второму классу — все остальные фигуры. После обучения ЭВМ соответствие машинной классификации

человеческой проверяется с помощью специальной *экзаменационной выборки.* Если полученный результат удовлетворяет человека, то считается, что ЭВМ, обучилась узнаванию правильно, Если же это не так, то ЭВМ использует экзаменационную выборку для уточнения своих решающих правил. Процесс обучения продолжается до тех пор, пока найденные решающие правила не будут давать устойчивую классификацию.

Обратим внимание на одну особенность, связанную с определением признаков объектов, проявившуюся в нашем примере и часто встречающуюся в самых различных задачах классификации и узнавания: не все признаки носят однозначный характер. Мы, конечно, предполагаем, что такие признаки, как цвет или форма, определяются сравнительно просто и однозначно. Но признак, связанный с размером объектов, более сложен. В самом деле, что значит "большой объект"? Где проходит граница между большим и маленьким объектами?

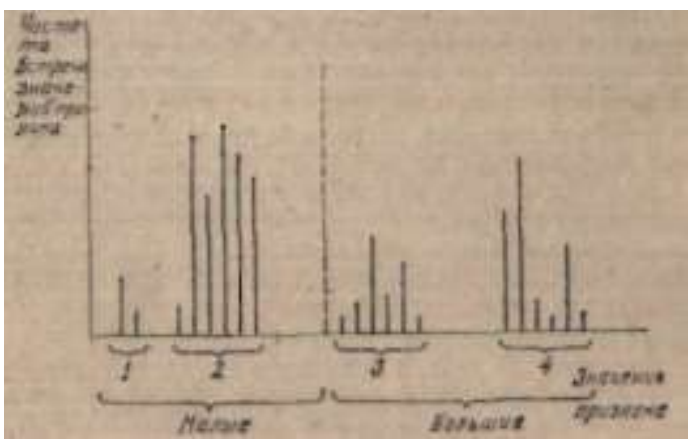



Рис. 8.

В программе М. М. Бонгарда эта трудность преодолевалась за счет специальной процедуры, названной им «Развал на кучи». Пусть мы имеем некоторый признак, принимающий различные значения. Отобразим частоту встречи значений признака так, как это показано на рис. 8. По горизонтали на рисунке отложены значения некоторого признака, а по вертикали — Число случаев, в которых встретились эти значения признака. Из рисунка, видно, что для показанного случая значения признака собраны в четыре группы, которые М. М. Бонгард назвал «кучами». Интересно отметить, что «развал на кучи» происходит без влияния на исход этого процесса семантики признака. Такой развал может произойти по признаку размера, но аналогичный развал может произойти и по признаку яркости, и по признаку веса, и вообще по любому другому Признаку. Если показанная на рисунке ситуация относится к значениям признака, определяющего размер объекта, то границу между малыми и большими значениями можно провести, например, так, как это показано на рисунке (хотя, конечно, эта группировка не единственная).

Сделаем еще одно замечание по процедуре М. М. Бонгарда, послужившей прообразом для многих других процедур классификации и узнавания принадлежности объектов к классам: эта процедура носит последовательный характер. Сначала происходит этап обучения, в процессе которого вырабатываются приемлемые решающие правила. Потом происходит работа узнающей системы. Эта работа состоит в выделении значении наблюдаемых признаков, образовании производных информативных признаков, сформированных в процессе обучения, и проверки удовлетворения тем или иным решающим правилам. Таким образом, процесс этот сугубо аналитический, последовательный. В то же время для человека и, по-видимому, для других живых организмов, наряду с таким аналитическим процессом узнавания, существует и другой, синтетический процесс, называемый в психологии симультанным узнаванием (одновременным, мгновенным). При реализации этого процесса мы узнаём объект как-то сразу, не вычленяя и не анализируя значения его наблюдаемых признаков. Вот вы встретили на улице человека. Едва взглянув на него, вы уже узнали его. Конечно, это ваш старый знакомый, с которым вы не виделись несколько лот! И вот уже начинается сбивчивый первоначальный диалог; «Привет, старик! Сколько лет, сколько зим! Как ты там?...».

Как же произошло в этом случае узнавание? Если проанализировать время, которое прошло с того момента, как вы взглянули на прохожего, до момента, когда

вы поняли, что это ваш старый приятель, и сравнить это время со скоростью обработки зрительной информа¬ции в мозгу (такие данные имеются в руках физиологов и психологов), то выяснится, что это время слишком мало для аналитической процедуры узнавания. За это время нельзя успеть выделить первичные признаки и, самое главное, последовательно обработать их для получения признаков, входящих в решающие правила. Объект в этой процедуре воспринимается человеком как бы одномоментно, интегрально. Возникает то, что психологи называют целостным образом объекта, его *гештальтом.* И этот гештальт как-то сравнивается с отпечатком прежних гештальтов, хранящихся в памяти.

Наличие двух типов процедур узнавания: аналитической и синтетической — явление весьма глубокого порядка. Как мы увидим дальше, это есть проявление фундаментального свойства человеческого мозга. Подобная двойственность пронизывает все стороны деятельности человека, присутствует в подавляющем числе интеллектуальных процедур.

Это означает, что наряду с дискретной переработкой информации мозг производит непрерывную обработку. Аналоги этому есть и среди вычислительных устройств. Кроме ЭВМ, суть которых связана с имитацией всевозможных дискретных процессов переработки информации, долгие годы существуют аналоговые устройства. В основном их используют для решения разнообразных дифференциальных уравнений. Но с их помощью можно решать и многие другие задачи, имитировать процессы, протекающие в технических устройствах, и делать многое другое. Принцип их действия в корне отличен от принципов, положенных в основу работы ЭВМ. Они моделируют физический процесс, суть которого аналогична изучаемому процессу. И время, затрачиваемое аналоговыми устройствами на это моделирование, есть время переходного электрического процесса, необходимое для окончательного установления значений электрических сигналов. Но кроме таких устройств, существуют и другие, работающие по иным, чем ЭВМ, принципам.

**Персептроны.** Огромные успехи, достигнутые в области решения задач на ЭВМ, на некоторое время оттеснили куда-то на периферию науки идеи о том, что



универсальная машина Тьюринга, воплощенная в архитектуре современных ЭВМ, далеко не единственный путь к имитации процессов, происходящих в живой и неживой природе. Поэтому столь неожиданным было появление в 1957 году обобщающей работы американского ученого Фрэнка Розенблатта, в которой он описал машину, совершенно отличную от ЭВМ. Он показал, что с ее помощью можно решать такие задачи распознавания и в такие короткие сроки, которые сравнимы с процессом симультанного узнавания, о котором мы говорили выше. И хотя, как потом выяснилось, устройства, предложенные Ф. Розенблаттом, во многом не оправдали возлагавшихся на них надежд, именно они показали, что путь построения машин, имитирующих интеллектуальные процедуры, не определяется однозначно. ЭВМ — не единственный способ построения таких машин, а возможно, и не самый эффективный.

Свои устройства Ф. Розенблатт назвал *персептронами.* Их основными элементами являются пороговые логические элементы. *Пороговый логический элемент* представляет собой устройство с $l$ входами и одним выходом. На входы устройства могут поступать двоичные сигналы. Для определения выходного сигнала, который тоже является двоичным, проверяется знак неравенства $a_1q_1 + a_2q_2 + \ldots + a_lq_l > \theta$, где $q_i$ — значения входных сигналов; $a_i$ — заранее заданные положительные числа называемые *весами;* $\theta$ — заранее заданное положительное число, называемое *порогом.* Если неравенство выполняется, то на выходе порогового элемента появляется единица. В противном случае его выход равен нулю.

Пусть имеется клеточное пространство, каждая клетка которого может находиться в закрашенном или незакрашенном состоянии. К каждой клетке может быть подсоединен любой вход любого порогового элемента из множества имеющихся в персептроне таких элементов. Если эта клетка закрашена, то на соответствующий вход поступает сигнал, равный единице. Сигнал от незакрашенной клетки равен нулю. Входы пороговых элементов распределяются по клеткам поля случайным равновероятным образом. Пусть имеется еще одно множество элементов, состоящее из суммирующих устройств. На их входы с помощью случайного равновероятного выбора подсоединены выходы порого¬вых элементов. На выходе сумматоров появляется чис-

ло, равное сумме единиц, поступивших на вход. Число сумматоров равно числу образов, которые мы хотим распознавать.

Персептрон функционирует следующим образом (рис. 9). Пусть, например, мы хотим отличать буквы *А* от букв *Б*. Изображения этих букв переводятся на клеточное пространство путем закрашивания необходимых клеток (на рисунке показано, как на клеточном пространстве изображена буква *А)*. После этого срабатывает слой пороговых элементов, сумматоры складывают приходящие на их входы единицы и выдают результаты в сравнивающий блок. В нашем примере, показанном на рис. 9, имеются десять пороговых элементов с $\theta = 1$ и два сумматора. Сравнивающее устройство узнаёт, на каком из сумматоров появилось максимальное значение. Этот сумматор определяет классифицируемое изображение как образ той буквы, к которой этот сумматор однозначно относится. На нашем рисунке классификация персептроном произведена правильно,

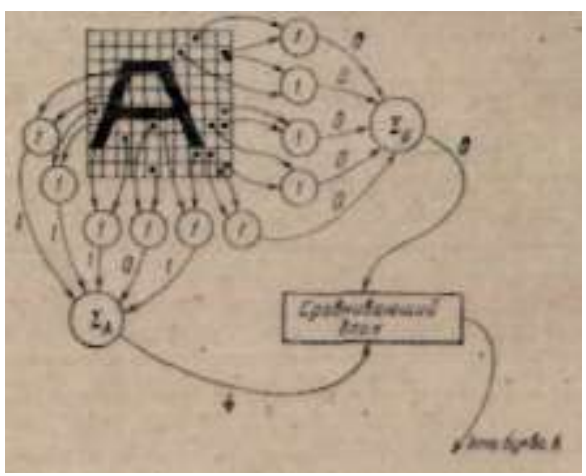



Рис 9.

Но при случайном соединении пороговых элементов с полями клеточного пространства и сумматорами нет никакой надежды на то, что персептрон будет правильно классифицировать изображения. Для того чтобы это произошло, необходим этап обучения персептрона. В процессе обучения можно, например, менять геометрию соединении пороговых элементов между собой,



но более эффективным приемом оказывается изменение значении весов на входах пороговых элементов или изменение значений их порогов. Изменять эти значения можно различными способами, например увеличивая веса полезных входов на некоторую фиксированную величину и уменьшая веса вредных входов. Такое линейное обучение не всегда приводит к цели. Вместо него можно ввести нелинейное изменение весов или порогов. Но как бы ни происходил этот процесс, персептрон может быть благодаря ему настроен на распознавание *А* и *Б*.

Идея персептронов в первый момент породила большой энтузиазм и надежды. В них усматривалась вторая составляющая интеллектуальных процессов — не алгебраическая, связанная с обработкой символов, а геометрическая, связанная с чертежом, изображенной или картиной. Персептроны Ф. Розенблатта стали эволюционировать. Появились новые промежуточные слои пороговых элементов, логика работы этих элементов, сумматоров и сравнивающего устройства стала усложняться. Был осуществлен переход от двоичных сигналов к сигналам, принимающим большее число значений. Однако все эти нововведения не привели к созданию новой универсальной модели. Пока что не удалось построить универсальный персептрон, способный имитировать работу любого реального персептрона. Да и тезиса о соответствии алгоритмов процессам, реализуемым в персептронах (даже потенциально, бесконечных по числу входящих в них элементов), пока никто еще не выдвигал.

Идея универсального персептрона, сверкнув яркой звездой на небе науки, тихо угасла. И лишь самые упорные ее сторонники продолжают искать пути для возрождения былой славы персептронов, ибо для практических нужд, связанных с обработкой изображений, персептроны в ряде случаев могли бы оказаться полезными и нужными.

Кроме идеи персептронов, периодически возникали идеи о создании других систем по переработке информации, структура и функционирование которых не были похожи на принципы, реализованные в ЭВМ. От них, пожалуй, остались лишь названия: пандемониум, артрон, матрица Штайнбуха и многие другие. Идеи эти продолжают свою жизнь и сейчас. Особенно после того,

как появились голография и оптоэлектроника. Голография и световоды сделали возможным передачу, хранение и обработку информации в виде различных картин. Новые вычислительные машины, построенные на этих принципах, реализуют идеи, отличные от машины Тьюринга. И в них, возможно, смогут быть объединены все те достижения, которые продемонстрировали нам «гадкие утята» предшествующих лет развития устройств для хранения и переработки информации.

Идею персептрона можно реализовать не только в специальном техническом устройстве. Ее можно воплотить и в программе, реализуемой на ЭВМ, Конечно, при этом теряются неоспоримые преимущества технической реализации: параллельность всех протекающих в персептроне процессов, симультанность процесса узнавания. Но программная модель на ЭВМ обладает рядом удобств, поскольку позволяет легко перестраивать структуру персептрона, менять число элементов в слоях пороговых элементов, менять правила функционирования персептрона. Большинство специалистов, занимавшихся персептронами, исследовали их с помощью подобной имитации.

Именно так поступил М. М. Бонгард, когда создавал свою узнающую программу «Геометрия». Выше мы описали процедуру классификации М. М. Бонгарда, которая давала имитацию аналитического узнавания. Программа «Геометрия» имитировала синтетическое узнавание.

Идея М. М. Бонгарда состояла в следующем. Пусть нам необходимо построить решающие правила, применение которых позволило бы классифицировать черно¬белые без полутонов картинки, разбивая их. на два класса. Поле, на которое проецируются картинки, имеет определенные размеры. В программе М. М. Бонгарда это поле имело размер 32 X 36 клеток. Процедура узна¬вания разбивается на четыре этапа, которым соответствуют четыре блока программы. Эти блоки имеют названия, ассоциирующиеся с теми, которые используются физиологами при описании тракта зрительного восприятия (М. М. Бонгард неоднократно подчеркивал поверхностный характер возникающей при этом аналогии). Сделано это для удобства сравнения действий программы с процессами, происходящими при узнавании объектов животными и человеком.

зе изображения.



Итак, четыре блока носят следующие названия: блок биполярных клеток, блок ганглиозных клеток, подкорка и кора. Они выполняют в программе следующие функции. Блок биполярных клеток, обрабатывая первоначальное изображение, порождает еще три изображения на поле 32 X 36: контуры фигур первоначального изображения, углы, имевшиеся у исходных изображений, и "вычищенные изображения", в которых убраны мелкие разрывы в контурах и отдельные закрашенные квадратики ноля, не связанные с изображениями. Соответственно этим задачам имеются три слоя специализированных биполярных клеток, которые осуществляют необходимые преобразования. Сами биполярные клетки могут быть реализованы в виде автоматов с небольшим числом состояний, объединенных в клеточную структуру по типу структуры Дж. фон Неймана.

Блок ганглиозных клеток в программе М. М. Бонгарда состоит из 45 слоев, каждый из которых включает 32 X 36 = 1152 пороговых элемента. Входы этих элементов соединены с четырьмя полями (исходным и тремя порожденными деятельностью биполярных клеток). Соединение ганглиозных клеток происходит случайным образом в процессе обучения. Но входы каждого порогового элемента (ганглиозной клетки) могут соединяться не со всем полом изображения, а только внутри квадрата — окна размером 8 X 8. Это позволяет устранять шум от ганглиозных клеток, получающих информацию от несвязанных частей изображения. Число входов ганглиозных клеток в программе «Геометрия» было выбрано равным 12. В пороговых элементах, используемых в описываемой программе, входы были двух типов: возбуждающие и тормозящие. Сигналы на возбуждающих входах от закрашенных полей были равны единице, а от незакрашенных полей — нулю. Сигналы на тормозящих входах от незакрашенных полей были равны нулю. Если же тормозящий вход был соединен с закрашенной клеткой изображения, то сигнал от него, независимо от значении остальных 11 входов ганглиозной клетки, давал на ее выходе нулевой сигнал. Выходом каждого ганглиозного слоя после его настройки с помощью изменения соединений и порогов является число от нуля до 1152, соответствующее количеству возбужденных клеток слоя при анали-

Подкорка также состоит из пороговых элементов. Она превращает выдаваемые ганглиозными слоями числа в нули или единицы в зависимости от того, превосходит ли это число значение порога элемента или нет. Пороги элементов подкорки подбираются в процессе обучения, когда обучение ганглиозных слоев уже завершено. И, наконец, кора строит по этой двоичной информации решающее правило — некоторую логическую функцию (формулу исчисления высказываний), с помощью которого оказывается возможным разделение классов объектов в обучающей выборке.

Работа с программой «Геометрия» показала, что ее функционирование сравнимо с функционированием персептрона с большим числом слоев пороговых элементов и что она обладает теми же недостатками, что и персептрон. Важнейшим из них является следующий. Пусть нам удалось произвести настройку элементов так, что происходит уверенное отличение буквы *А* от буквы *Б,* Пусть теперь на поле рецепторов нанесено изображение *БА.* Для человека ясно, что здесь присутствуют два уже известных ему объекта: *А* и *Б.* Но для персептрона или программы «Геометрия» это совсем не так. Они лишены аналитических процедур, позволяющих разлагать изображения на части. Поэтому изображение *БА* они воспринимают как объект нового класса, не имеющий ничего общего ни с объектом, относящимся к классу *А,* ни с объектом, относящимся к классу *Б.* В этом и состоит основной недостаток персептронов и аналогичных им систем и программ. Ибо анализ и синтез — две неразрывные стороны одного процесса познания, частным случаем которого является процесс классификации и узнавания. Моделирование реальных интеллектуальных процессов должно включать в себя имитацию обоих этих компонентов.

## 2.4. Краткое резюме

История накопила много различного материала для развития теории искусственного интеллекта. Мы постарались отметить основные вехи на этом пути. Попробуем теперь в сжатой форме сформулировать основные результаты.

1. Среди всех процедур может быть выделен весьма широкий класс алгоритмических процедур, для которых



можно создать формальную модель процесса и построить теоретическую модель реализации этих процедур на универсальном устройстве (например, в виде машины Тьюринга).

2. Реальные технические устройства могут обладать практической универсальностью при реализации алгоритмических процедур, т. е. могут реализовать любую из процедур с заданным объемом необходимой памяти. В частности, такими устройствами являются ЭВМ, сохраняющие традиционные принципы переработки информации.

3. Существуют такие классы технических устройств и такие теоретические модели (клеточные среды, клеточные автоматы, персептроны и многие другие), которые функционируют на основе принципов, отличных от машины Тьюринга. Эти устройства эффективно реализуют некоторые процедуры, которые представляют при имитации на ЭВМ весьма значительные трудности. Однако универсальность подобных устройств не доказана.

4. Имитация таких свойств, присущих живым организмам и человеческому поведению, как условно-рефлекторная деятельность, стильно-реактивная деятельность, адаптация, самоорганизация, целесообразное поведение в условиях неполной информации, размножение путем самокопирования, конструирование по описанию, узнавание и распознавание, может происходить с помощью различных технических средств небольшой сложности.

5. ЭВМ способны моделировать такие формы интеллектуальной деятельности, как решение математических задач, сочинение произведений искусства, игру в различные игры и т. п. При этом необходимо помнить, что, по существу, эта деятельность выполняется не одной ЭВМ, а парой: программист— ЭВМ.

6. Существуют два пути имитации свойств живых организмов и интеллектуальной деятельности человека: структурно-функциональный и информационный. При первом подходе мы строим структуру технического устройства и организуем ею функционирование так, чтобы это помогло имитировать необходимый процесс. Примером такого подхода может служить модель персептрона. При втором подходе мы стремимся создать устройство, способное работать с описанием того про-

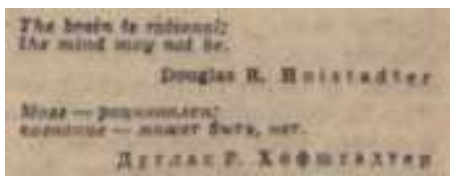
цесса, который мы собираемся имитировать. Сам процесс при этом имитируется только на информационном, а не на физическом уровне. Этот подход реализован в современных ЭВМ. Какой из этих двух подходов предпочтительнее — пока неизвестно. Возможно, что оба они необходимы.

7. Недостатком всех программ моделирования человеческой деятельности и структурно-функциональных устройств для этого является их узкая специализация. По существу, для каждой новой задачи надо создать новую программу для ЭВМ или строить новое техническое устройство, структура и функционирование которого определяются решаемой задачей.

Эти положения позволят читателю оценить то новое, с чего начнется следующая глава книги.

ГЛАВА ТРЕТЬЯ

# СОВРЕМЕННОСТЬ




## 3.1. Метапроцедуры

**Связка ключей или отмычка?** Мы уже говорили о первых успехах составления программ для ЭВМ, с помощью которых удалось имитировать с использованием машинных процедур решение задач, считавшихся ранее прерогативой человеческого интеллекта. Покажем теперь, что эти успехи мало что значат при решении проблем создания интеллектуальных систем.

Пусть в нашем распоряжении имеется весьма мощ¬ная современная ЭВМ. И пусть мы составили для нее прекрасную программу для игры в русские шашки. Программа эта полна, осуществляет в приемлемое время анализ всех продолжений шашечной игры из данной позиции. Она обеспечивает ЭВМ беспроигрышную игру. Можем ли мы считать, что пара ЭВМ — шашечная программа образует интеллектуальную систему? По-видимому, найдется немало скептически настроенных читателей этой книги, которые не согласятся с тем, чтобы признать такую ЭВМ интеллектуальной. Добавим к шашечной программе еще какую-либо высококачественную программу, имитирующую другой вид интеллектуальной творческой деятельности человека, например программу сочинения мелодий несложных музыкальных произведений. Теперь наша ЭВМ не только играет в русские шашки, но и выполняет работу, доступную далеко не всем людям, — сочиняет вальсы и марши несложной структуры. Можно ли те-

перь считать эту ЭВМ вместе с введенными в нее программами интеллектуальной системой?

По-видимому, снова найдутся скептики, которые посчитают, что этого еще слишком мало. Добавим очередную «интеллектуальную» программу, сделав ЭВМ способной к доказательству теорем математической логики. И снова зададим свой традиционный вопрос об интеллектуальности возникшей системы.

Описанную нами процедуру можно повторять сколь угодно долго. На каждом шаге ЭВМ будет приобретать способность к выполнению еще одного вида деятельности, относимого традиционно к интеллектуальному. Часть экспертов на каких-то шагах, возможно, признает ЭВМ, снабженную таким набором программ, интеллектуальной системой. Но всегда найдутся и такие, которые укажут десятки видов интеллектуальной деятельности, еще недоступных нашей ЭВМ.

Эта ситуация побудила Л. Тьюринга сформулировать свой знаменитый тезис об определении того момента, начиная с которого искусственную систему следует считать интеллектуальной. Идея этого тезиса состоит в следующем. Пусть эксперт ничего не знает о наборе программ, заложенных в ЭВМ. Более того, его общение с ЭВМ происходит не непосредственно, а с помощью телефона или какого-либо другого устройства дистанционной связи (например, телетайпа), которым могут пользоваться как ЭВМ, так и человек. Во время экспертизы человек не знает, с кем он общается. На другом конце канала связи может быть либо ЭВМ, либо человек. Эксперт задает любой набор тестов-заданий. На некотором шаге экспертизы он должен определить, является ли его «собеседник» человеком или ЭВМ, снабженной набором специальных программ. Если в условиях подобного эксперимента эксперт ошибочно принимает ЭВМ за человека, то, согласно тезису Тьюринга, такую искусственную систему следует считать интеллектуальной.

Тезис Тьюринга можно принимать или не принимать. Это дело вкуса. Но для многих кибернетиков он стал точным мерилом того, что можно назвать интеллектуальной системой. Казалось, что Тьюринг на долгие годы поставил перед теми, кто стремился построить интеллектуальную систему, трудновыполнимую задачу.



Но произошло иначе. В конце шестидесятых годов американский кибернетик Дж. Вейценбаум написал две программы, названные им ЭЛИЗА и ДОКТОР. Цель этих программ—сбить с толку эксперта в условиях теста Тьюринга. В их основе лежит идея так называемого *фатического диалога.* При фатическом диалоге имеется только видимость беседы. Оба собеседника или один из них, по существу, не вникает в смысл сообщаемого ему, а строит свой ответ на основе чисто формального преобразования услышанного. Вот пример такого диалога, реализованного программой ЭЛИЗА в одном из экспериментов Дж. Вейценба¬ума:

«Человек: Вы не очень настойчивы, но я думаю, что Вы не хотите, чтобы я это заметил.

ЭЛИЗА: Что заставляет Вас думать, что я не очень настойчива?

Человек: Вы не спорите со мной.

ЭЛИЗА: Почему Вы думаете, что я не спорю с Вами?

Человек: Вы боитесь меня.

ЭЛИЗА: Доставляет ли Вам удовольствие считать, что я Вас боюсь?».

Такой диалог может продолжаться сколь угодно долго. ЭЛИЗА с помощью встроенных в нее стандартных средств перефразирует то, что сообщает ей человек, превращая его утверждения в вопросы, а вопросы— в соответствующие им ответы-утверждения. Эксперименты с ЭЛИЗОЙ и последовавшей за ней программой ДОКТОР показали, что часть экспертов считает, что диалог происходит не с ЭВМ, а с человеком. С точки зрения тезиса Тьюринга это означает, что снабженная этими программами ЭВМ является интеллектуальной системой.

Несколько позже произошел курьезный случай, когда вице-президент крупной американской компании BBN по производству ЭВМ, сам того не ведая, реализовал тест А. Тьюринга.

Как-то ему понадобилось связаться с известным системным программистом Д. Бобровым, работающим в этой компании. Стандартная связь по телетайпу привела к следующей беседе:

«Вице-президент: Нельзя ли мне сейчас воспользоваться ЭВМ?

Бобров: А почему Вы спрашиваете?

Вице-президент: Может быть, мне удастся заключить сделку.

Бобров: Почему Вы в этом уверены?

Вице-президент: Мои предполагаемые покупатели никогда не видели нашу систему в действии.

Бобров: Это имеет для Вас большое значение?

Вице-президент: Конечно.

Бобров: Вы как будто сомневаетесь в этом?

Вице-президент: Что за нелепое замечание?

Бобров: Что Вы имеете в виду, говоря о нелепом замечании?

Вице-президент: Возражаете ли Вы против того, чтобы я сейчас воспользовался ЭВМ?

Бобров: Расскажите мне побольше о Ваших предполагаемых покупателях.

Вице-президент: Позвоните мне 491-1850».

Вице-президент был, по-видимому, обескуражен таким странным разговором с ведущим программистом. Еще более странным было то, что звонка от Д. Боброва не последовало. Недоразумение выяснилось лить позднее, когда вице-президент стал отчитывать своего сотрудника. Оказалось, что телетайп соединил его не с самим Д. Бобровым, а с ЭВМ, на которой была реализована экспериментальная диалоговая программа, реализующая подобно ЭЛИЗЕ фатический диалог.

Эти примеры показывают, что тезис Тьюринга оказался несостоятельным.

Вернемся к обсуждению нашего мысленного эксперимента с ЭВМ, в память которой мы заложили некоторый набор программ, позволяющих машине имитировать определенные виды интеллектуальной деятельности. Отметим одну особенность. Программы, записанные в памяти ЭВМ, узко специализированы, предназначены для выполнения только одного вида конкретной деятельности, изолированы друг от друга. То, что ЭВМ способна доказывать теоремы математической логики, никак не помогает ей играть в шашки. А умение обыгрывать своих противников в эту игру, ничуть не увеличивает способности ЭВМ в сочинении вальсов. Другими словами, накопление программ в памяти ЭВМ не приводит к качественным изменениям в ее возможностях и ничего не дает с точки зрения



овладения иными, чем предусмотрено в этих программах, видами интеллектуальной деятельности. Этот набор программ напоминает большую библиотеку. В каждой из книг, хранящихся в ней, содержатся какие-то конкретные сведения. Но добавление новой книги в книгохранилище никак не сказывается на этих сведениях и не увеличивает «интеллекта библиотеки».

Каждая программа, имитирующая определенный вид интеллектуальной деятельности, может быть уподоблена ключу, предназначенному для открывании определенной двери. Связка таких ключей может оказаться абсолютно бесполезной при попытке открыть дверь, замок которой имеет новую природу.

Вместо таких индивидуальных ключей лучше было бы иметь набор универсальных отмычек, позволяющих открыть любой возможный замок в дверях. Другими словами, вместо программ (даже очень хороших), предназначенных для имитации конкретных видов интеллектуальной деятельности, было бы желательно иметь программы, имитирующие некоторые общеинтеллектуальные процедуры *(метапроцедуры).*

**Лабиринтная гипотеза.** Вспомним опыты Э. Торндайка с крысами в лабиринте и результаты в области адаптации автоматов, полученные М. Л. Цетлиным, его учениками и последователями. Эти исследования и результаты, полученные при программирования игр (обход дерева игры с отсечением за счет эвристик отдельных поддеревьев — ведь это тоже блуждание по лабиринту), позволили сформулировать гипотезу о существовании метапроцедуры, присутствующей при решении многих интеллектуальных задач. Эта метапроцедура есть целенаправленный поиск в *лабиринте возможностей.*

Процедура эта является весьма «машинной», легко воплощаемой в программах. Именно поэтому она стала первой из метапроцедур, значение которой было осознано специалистами по имитации интеллектуальных процессов. Но, к сожалению, для многих из них она осталась единственной и универсальной. Некоторые крайние сторонники перебора и поиска в лабиринте возможностей так и говорили: «Когда ЭВМ станут еще более мощными, все задачи можно будет решать перебором, ибо перебор универсален».

Логическим завершением идеи лабиринтной гипотезы стали универсальные программы для ЭВМ, предназначенные для решения "любых задач". И первой среди них была программа, «скромно» названная ее создателями "Общий решатель задач". Она была создана в США к началу шестидесятых годов. Ее авторами были кибернетики А. Ньюэлл и Дж. Шоу и известный психолог Г. Саймон. В названии ее подчеркивалась универсальность процедуры, реализованной в программе. Она не была ориентирована на решение какой-либо одной конкретной задачи. Авторы программы надеялись, что «Общий решатель задач» сможет решать любые задачи, если начальные условия их и цели будут сформулированы на языке, доступном программе.

Для нас будет полезно на некотором примере разобраться в специфике «Общего решателя задач».

Пусть в нашем распоряжении находится множество различных объектов. И пусть к этим объектам мы можем применять $m$ различных преобразований. Объекты мы будем обозначать символами $a_i$, а преобразования — символами $d_i$. Если к объекту $a_i$ применено преобразование $d_j$ то результатом его будет новый объект $a_k$ (в частном случае новый объект может совпадать с исходным объектом $a_i$). На множестве объектов введено понятие различия. Для каждой пары объектов $(a_i, a_j)$ можно установить, какими различиями из заданного списка $r_1, r_2, ..., r_q$ они обладают. Основным компонентом "Общего решателя задач" является таблица различий. Эта таблица имеет столько столбцов, сколько различий фиксируется между $a_i$. Число строк этой таблицы равно числу различных преобразований $d_j$. Клетки таблицы либо могут быть пустыми, либо в них может стоять знак «+». Этот знак указывает на тот факт, что соответствующее преобразование может использоваться для устранения (или уменьшения, если различия как-то оцениваются по величине) того различия, которым помечен соответствующий столбец. Различия могут быть ранжированы по важности, но этого может и не быть.

Работа программы состоит в сравнении исходного элемента и целевого элемента. Если между ними нет различий, то задача решена. Если между ними наблюдаются различия, то на основании таблицы различий программа определяет те преобразования, которые могут повлиять на устранение одного из различий. Произвольно выбирается одно из таких преобразований и применяется к исходному объекту. Новый объект сравнивается с целевым. Если между ними нет различий, то решение найдено. В противном случае повторяется описанная выше процедура.



Из этого описания следует, что в «Общем решателе задач» процедуры организованы таким образом, что семантика самой задачи никак не влияет на них. Нужно только так переформулировать исходную задачу, чтобы все дело свелось к поиску различий между исходной ситуацией и целевой и применению преобразований из фиксированного списка. В этом будет состоять подготовка задачи к решению. Кроме того, для задач из данной проблемной области нужно выбрать необходимые различия и преобразования и составить таблицу различий. После этого «Общий решатель задач» превратится в конкретизированную программу, нацеленную на решение задач из определенного класса.

В качестве первой конкретизированной программы авторы «Общего решателя задач» рассмотрели программу, предназначенную для доказательства теорем в исчислении высказываний.

В исчисления высказываний все теоремы состоят в доказательстве или опровержении формул вида $f_1 = f_2$, где $f_1$, и $f_2$ — сложные высказывания, записанные в символическом виде с применением знаков операций исчисления высказываний (отрицание, конъюнкция, дизъюнкция и импликация). Две формулы считаются в исчислении высказываний равными, если оно при любых значениях входящих в них элементарных высказываний интерпретируются с точки зрения истинности и ложности одинаково. Рассмотрим, например, два таких высказывания: «Человек не может быть одновременно лгуном и всегда говорящим правду» и «Человек или не лгун, или не всегда говорит правду». Интуиция подсказывает нам, что эти два высказывания одинаково интерпретируются с точки зрения истинности и ложности.

Более удобно от словесных формулировок высказывании перейти к их символической записи. Если $a$ и $b$ суть элементарные высказывания «Человек лгун» и «Человек всегда говорит правду», то приведенные выше сложные высказывания могут быть представлены в

следующей форме: $f_1 = \underline{ab}$ и $f_2 = a \vee b$. Нас интересует, справедливо или несправедливо равенство $f_1 = f_2$. Стандартный прием такой проверки заключается в подстановке в формулы вместо $a$ и $b$ всех возможных значений истины и лжи с проверкой совпадения истинности и ложности формул. Если они совпадают на всех возможных наборах для $a$ и $b$, то $f_1 = f_2$. В противном случае равенства нет. Проверку наших двух формул мы сведем для удобства в таблицу, в которой последовательно строятся интерпретации обеих формул (для этого используется интерпретация основных логических операций); выделенные двойными линиями столбцы соответствуют сравниваемым формулам (табл. 1):

Таблица 4

| $a$ | $b$ | $ab$ | $\overline{ab}$ | $\overline{a}$ | $\overline{b}$ | $a \vee b$ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 | |
| 0 | 1 | 0 | 1 | 1 | 0 | |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 |

Совпадение выделенных в этой таблице столбцов доказывает равенство $f_1 = f_2$.

Способ доказательства теорем при помощи составления таблицы всех возможных интерпретаций формул исчисления высказываний весьма легко поддается программированию и реализации на ЭВМ. Но у него есть один крупный недостаток. С ростом числа элементарных высказываний, входящих в формулу, число строк таблицы возрастает в $2^n$ раз ($n$ — число элементарных высказываний). Поэтому быстро увеличивается время, необходимое ЭВМ для выполнения проверок. Кроме того, сама эта процедура слишком мало похожа на способы "человеческого" доказательства утверждений в математике и, в частности, в математической логике.

В "Общем решателе задач" используется иной подход к доказательству теорем исчисления высказываний. Идея этого подхода состоит в том, что при доказательстве равенства $f_1 = f_2$ (или опровержении его)



ищется цепь элементарных преобразовании, которая либо $f_1$ преобразует в $f_2$, либо $f_2$ преобразует в $f_1$, либо приводит путем преобразований $f_1$ и $f_2$ к одной и той же формуле $f_3$. Если программа убеждается, что равенства достичь невозможно, то принимается утверждение $f_1 \neq f_2$.

На каждом шаге преобразований левая и правая части равенства сравниваются между собой. Если между ними нет различия, то теорема доказана. Если же различие есть, то выясняется его характер. Для данной конкретной задачи в "Общем решателе задач" выявляются семь типов различий между формулами. Во-первых, формулы могут отличаться символами элементарных высказываний, которые в них входят, или числом вхождении некоторого высказывания в формулу. Примеры таких различий демонстрируют следующие две пары формул: $a$ и $a \vee b$, а также $a$ и $a \vee a$. Во-вторых, формулы могут отличаться по набору операций, используемых в них, как это было в формулах предшествующего примера: $ab$ и $a \vee b$. Другими типами различий являются: различие в общем отрицании над формулой (например, $a \vee b$ и $\overline{a \vee b}$), различие в группировке символов в формуле [например, $a(bc)$ и $(ab)c$], различие в порядке следования символов (например, $a \vee b$ и $b \vee a$), различие в наличии отрицания над подформулой (например, $a \vee bc$ и $a \vee \overline{bc}$).

Это и есть те семь видов различий, которые используются при доказательстве теорем (именно семь, так как первый пункт содержит два вида различий). Для устранения этих различий к формулам применяются преобразования, разбитые в описываемой программе на 12 типов. Не будем их все перечислять, а также строить таблицу различий, имеющую размер 7 X 12. Построим лишь фрагмент этой таблицы. Рассмотрим, например, преобразование $A(B \vee C) \Leftrightarrow (A \vee B)(A \vee C)$. Двойная стрелка означает, что преобразование можно применять в любом порядке: справа налево или слева направо. Формулы, стоящие слева и справа от двойной стрелки, равны друг другу. В этом можно убедиться, применив описанный выше способ проверки их совпадения при всех возможных интерпретациях. Заглавные буквы в преобразовании означают, что вместо них могут стоять не только элементарные высказывания, но и произ-

вольные формулы. Что меняет подобное преобразование в формуле, к которой оно применено (для определенности считаем, что преобразование применяется слева направо)? Во-первых, оно увеличивает на единицу число вхождений подформулы $B$ в преобразуемую формулу. Во-вторых, оно меняет группировку членов в формуле. В-третьих, оно меняет внешнюю и внутреннюю операции в формуле так, что порядок выполнения операций становится иным. Это означает, что в таблице различий в строке, которой соответствует рассматриваемое преобразование; будут стоять знаки «+» в столбцах, соответствующих указанным различиям в формулах (для принятого нами порядка описания различий знак «+» будет поставлен во втором, третьем (так как прежде всего анализируется различие во внешней операции) и пятом столбцах. Другим примером преобразования может служить правило $A \Leftrightarrow \Leftrightarrow AB \vee AB$, позволяющее вводить новый символ в формулу и, следовательно, применимое для устранения первого различия.

Когда таблица различий составлена и программа приступает к работе, то при решении конкретной задачи она сталкивается с некоторыми проблемами. Первой из них является неоднозначность ее действий. В паре формул $f_1$ и $f_2$ могут одновременно быть различия разного типа. Например, для формул $a \rightarrow b$ и $a \vee b$ имеются различие в виде операции и различие в наличии отрицания над подформулой. Какое из них более важно? Что должна делать прежде всего программа? По мнению одного из авторов «Общего решателя задач» Г. Саймона, человек в подобной ситуации всегда ранжирует различия, считая одни из них более существенными, а другие не столь важными. Для выявления «человеческих» предпочтений при доказательстве теорем исчисления высказываний Г. Саймон провел многочисленные эксперименты с людьми, решавшими аналогичные задачи. На основании обработки результатов экспериментов (люди в процессе поиска или опровержения соотношений $f_1 = f_2$ проговаривали вслух все свои соображения по поводу совершаемых ими действий) авторы программы выявили ранжировку различий. Оказалось, например, что наиболее важным различием люди считают несовпадение символов в формулах по их виду и числу вхождений, а разницу в



группировке или порядке следования символов учитывают в самую последнюю очередь. Поэтому различия в таблице различий расположены в строгом порядке по убыванию их значимости. И программа стремится прежде всего ликвидировать наиболее весомое из имеющихся различии.

Второй проблемой для программы является неоднозначность выбора преобразования для ликвидации обнаруженного различия. Какое преобразование она должна выбрать? Для устранения этой трудности авторы «Общего решателя задач» поступили следующим образом. Они выдвинули принцип максимального сохранения достигнутого. Согласно этому принципу, наилучшим преобразованием на каждом шаге является то, которое максимально устраняет рассматриваемое различие и не портит того, что уже было достигнуто ранее. Если, согласно этому принципу, несколько преобразований равноценны, то равновероятно выбирается любое из них.

Третья проблема заключается в том, что движение по лабиринту возможных преобразований эвристично. Другими словами, мы не гарантированы, что не наступит момент, когда никакие преобразования не смогут уменьшить имеющиеся различия и цель окажется недостижимой из-за того, что где-то ранее мы сделали не слишком хороший выбор. Правда, при доказательстве теорем исчисления высказываний всегда можно вернуться назад (все преобразования в этой задаче обратимы), но в общем случае это утверждение сохраняет свою силу. Кроме того, как программа узнаёт, что на самом деле $f_1 \neq f_2$? Когда она должна прекратить поиск доказательства? На этот вопрос ответа нет. В реальной программе поиск прекращается после определенных затрат времени ЭВМ, которые считаются авторами программы вполне достаточными для доказательства любой «разумно сложной» теоремы.

Определенный успех «Общего решателя задач» при доказательстве теорем исчисления высказываний вдохновил ее создателей. Они широковещательно заявили, что ими найдена та универсальная метапроцедура, которая поможет имитировать с помощью ЭВМ решение большинства интеллектуальных задач. Проверить это А. Ньюэлл, Дж. Шоу и Г. Саймон решили на шахматах.

Шахматный лабиринт огромен, но число различных преобразовании (элементарных ходов) не слишком велико. Оставалось сформулировать набор различий в шахматных позициях и заполнить таблицу различий. Авторы «Общего решателя задач» сделали и это. Но программа к их удивлению функционировала весьма плохо. Она делала элементарные ошибки и просчеты. Почему это произошло? Прежде всего потому, что различия текущих позиций от конечных (матовых) не могут быть сформулированы достаточно разумно. Поэтому при использовании «Общего решателя задач» для шахматной игры, были введены не различия между текущей и конечной целевыми ситуациями, а между парами соседних локальных ситуаций. Но локальные улучшения при просмотре на небольшое количество полуходов вперед, как мы уже говорили, не могут привести к сколько-нибудь удовлетворительно играющей машинной программе.

Неудача «Общего решателя задач» при игре в шахматы привела к тому, что его создатели ввели в свою программу специальный уровень глобального планирования. Задачей этого уровня является анализ лабиринта возможностей, «глядя на него сверху». Этот уровень должен был наметить общее направление движения к цели, а затем прежняя программа, пользуясь таблицей различий и информацией, полученной от уровня планирования, должна была осуществлять локальные перемещения по лабиринту в нужном общем направлении. Эта идея, верная в принципе, в «Общем решателе задач» не могла быть доведена до логического завершения из-за приверженности авторов программы к идее чистого поиска в лабиринте.

Трудности, связанные с применением «Общего решателя задач», можно проиллюстрировать следующим наглядным примером. В 1751 году английский музыкант Уильям Хейс написал руководство по сочинению музыки, которое по бытовавшей тогда традиции имело весьма пространное название: «Искусство сочинять музыку исключительно новым методом, пригодным для самых захудалых талантов». В этом сатирическом трактате Хейс предлагал следующий оригинальный способ сочинения музыкальных произведений. Взяв чистый лист нотной бумаги и положив его на стол, надо окунуть обыкновенную сапожную щетку в тушь и стряхнуть тушь со щетки на нотный лист. После этого, пишет Хейс, ваше музыкальное произведение уже есть на нотном листе. Надо только удалить с него все лишние кляксы. Возможно, что Хейс и прав. Вы могли так расположить кляксы на нотных строках, что после вычищения всего ненужного останется мелодия, приятная для слуха. Но, может быть, это и не так. Кто знает? Перед вами лабиринт возможных вычищений. И можно попробовать применить к нему идеи «Общего решателя задач». Но формирование различий и составление таблицы различий при такой постановке задачи вряд ли приведут вас к цели.



**Модельная гипотеза.** Неудачи «Общего решателя задач» наглядно продемонстрировали, что с помощью только метапроцедуры целенаправленного поиска вряд ли можно решать любые задачи. Нужны еще какие-то другие метапроцедуры.

Пусть перед нами на столе лежат шесть спичек. Задание гласит: «Не ломая и не сгибая спички, сложить из них четыре одинаковых равносторонних треугольника». Если вы не знаете решения этой задачи, то не поленитесь, достаньте спички и попробуйте ее решить, а потом уже читайте дальнейший текст.

В задаче о спичках лабиринт возможностей задан тем обстоятельством, что спички положены на поверхность стола. Перемещая их всевозможными способами по поверхности стола, можно довольно быстро придти к убеждению, что решение поставленной задачи найти в этом лабиринте перемещений невозможно. Что же делать? Те люди, которые не откажутся от поиска решения, через некоторое время поймут, что решение поставленной задачи существует в трехмерном пространстве. Надо сложить треугольник из трех спичек на поверхности стола, а с помощью оставшихся трех спичек построить тетраэдр — пирамиду, основание и три треугольные грани которой образуют четыре треугольника, составляющих цель задачи.

Обратим внимание на тот момент в решении задачи, когда лабиринт на поверхности стола привел нас к пониманию невозможности найти решение при поиске в нем. Что произошло дальше? Психологи называют этот момент *инсайтом (озарением)*. Инсайт — это переход от одного лабиринта возможностей к другому, построение нового лабиринта, в котором решение на-

ходится. Во время инсайта человек испытывает сильные положительные эмоции, радость по поводу найденного решения. И тот, кто испытывая это состояние, никогда не спутает его ни с каким другим.

О том, что именно *построение лабиринта,* в котором можно организовать эффективный поиск решения, является центральной метапроцедурой творческой деятельности, впервые четко сказал и обосновал экспериментально крупнейший представитель советской психологии В. Н. Пушкин. Он прожил недолгую, но яркую жизнь ученого, который следованию традиционным и устоявшимся научным представлениям предпочитал не всегда обоснованные и рискованные формулировки новых концепции и теорий. Во время панихиды один из руководителей института, где работал В. Н. Пушкин, очень точно сказал о нем: «Ум его был эвристичен». Его идеи, щедро рассыпаемые вокруг, служили многим специалистам благодатной темой для проведения фундаментальных исследований.

Но В. Н. Пушкин не только указал, что построение лабиринта является важной метапроцедурой, он еще выявил и суть механизмов, лежащих в ее основе. Эти механизмы были найдены им и его учениками с помощью серии блестяще поставленных опытов с людьми, решающими задачи лабиринтной природы.

Опишем суть одного из подобных экспериментов. Многим хорошо известна математическая игра, называемая «Игра в 15». На поле, имеющем 16 клеток, в определенном порядке расположены 15 фишек, на которые нанесены номера 1,2, …, 15. Одна клетка остается свободной и используется для перемещения фишек. Задача состоит в поиске такого перемещения, которое из начального расположения фишек приводит в некоторое другое, заранее заданное. Лабиринт перемещений, связанный с этой игрой, слишком велик для проведения экспериментов. Поэтому В. Н. Пушкин рассмотрел усеченный вариант игры, названный им «Игра в 5». Поле в этой игре имеет 6 клеток, на которых располагаются 5 фишек с номерами 1,2, …, 5. Задача участника эксперимента состоит в преобразовании заданного начального расположения фишек в заданное конечное их расположение.

Если бы для решения этой задачи был привлечен «Общий решатель задач», то потребовалось бы описать те преобразования и различия, на основе которых поставленная задача может быть решена. Преобразованием на каждом шаге является перемещение одной фишки на свободную клетку. Если она угловая (поле имеет размеры 2X3), то на нее может быть передвинута одна из двух соседних фишек. Если Же свободной является одна из двух средних клеток, поля, то на нее может быть передвинута одна из трех соседних фишек. Различие в текущем расположении фишек и целевым (конечным) расположением можно подсчитать, например, по числу инверсий, имеющихся в этих последовательностях. Так, в последовательностях 12354 и 12345 есть одна инверсия, а в последовательностях 23415 и 12345 число инверсий равно трем. Другими словами, число инверсий показывает, сколько соседних парных перестановок необходимо сделать, чтобы привести одну последовательность к другой. Для работы «Общего решателя задач» можно было бы рекомендовать выбор того преобразования из возможных на данном шаге (точнее, не одного преобразования, а группы их), которое максимально уменьшает число инверсий между получаемым в результате преобразования набором и целевым набором.



Но в экспериментах В. Н. Пушкина эта информация об инверсиях испытуемым не сообщается. В процессе своей деятельности они должны выработать собственную систему различий. Они должны сформировать тот лабиринт, в котором они будут искать решение, совершая целенаправленные преобразования.

При экспериментах по зрительному восприятию специалисты давно установили одну особенность, связанную с движением глаз при анализе зрительных ситуаций (например, при рассматривании картин или чтении написанного текста). Глаз человека не движется по изображению равномерно. Он как бы скачет по изображению, фиксируя на определенные моменты времени наиболее значительные или интересные его места. Существует несколько методик регистрации движений и фиксаций глаза. Группой, руководимой В. Н. Пушкиным, использовалась для этих целей, как правило, кинорегистрация движений. Пусть, например, испытуемый осматривает позицию, возникшую на шахматной доске. Для кинорегистрации доску помещают на некотором расстоянии от испытуемого, го-

лова которого зафиксирована специальным подбородником. В одном из черных полей доски сделано небольшое отверстие, через которое киноаппарат снимает движение глаза. По команде экспериментатора испытуемый, который до этого сидел с закрытыми глазами, открывал глаза и начинал осматривать шахматную позицию. Киноаппарат производил непрерывную съемку. Потом пленка обрабатывалась и выявлялись точки фиксации глаза. Перемещения глаза от одной точки фиксации до другой заменялись отрезками прямых. Получавшаяся после этого ломаная линия характеризовала динамику осмотра позиции. Аналогичным образом происходила и кино регистрация движения глаз при анализе позиций в «Игре в 5». Только в этом случае испытуемым одновременно предъявлялись две позиции: начальная и целевая. Обе позиции располагались в экспериментах одна под другой в одном поле зрения.

Все движения глаза (отметим, что во время скачка глаз ничего не видит) вместе с точками фиксации можно разделить на три типа: движения по исходной позиции, движения по целевой позиции и движения перехода от осмотра одной позиции к другой. На рис. 10 показана одна из кинорегистраций этого процесса (кружочками отмечены точки фиксации глаза). При этом необходимо помнить, что точка фиксации определяет только центр зрачка. Испытуемый видит при этом и другие фишки, а так называемым боковым зрением видит что-то и вне игрового поля. Однако основное его внимание сосредоточено в зоне фиксации центра зрачка. На рис. 10 показана только малая часть движений глаза при анализе исходной в целевой позиций. Испытуемые многократно переходят от осмотра одной позиции к другой и фиксируют глаз на одних и тех же клетках игровых полей. Но качественную сторону процесса осмотра позиций рисунок передает верно.

Какие выводы были сделаны психологами на основе анализа множества подобных пар позиций? Были рассмотрены два показателя: успешность решения испытуемым тестового набора задач, возникающих при «Игре в 5» (т. е. при различных наборах пар исходных и целевых позиций), и количество движений, совершаемых в процессе анализа этих пар позиций. Между этими двумя показателями обнаружилась вполне устойчивая обратная зависимость. Чем успешнее человек решал задачи, тем меньше тратил он времени на осмотры позиций, особенно на движения между позициями. С другой стороны, давно известно, что во многих видах управленческой диспетчерской деятельности человека наблюдается подобная же обратная зависимость между качеством той модели управления, которой пользуется диспетчер (уровнем ее сформированности), и количеством его информационных запросов. Большая информационная работа, объективно регистрируемая в движениях глаз, позволяет испытуемому найти путь решения задачи. И чем больше этих движений (особенно между позициями), тем с большими трудностями решает человек возникшую перед ним задачу, тем хуже сформирован лабиринт, в котором ему необходимо совершать поиск.

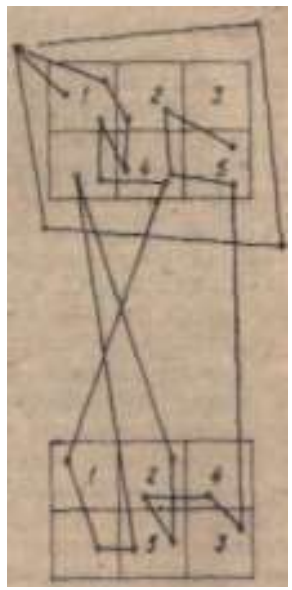


Рис 10.

Не менее интересны результаты кинорегистрации движения и фиксации глаз при осмотре шахматной позиции. Особенно разительно отличие в общей картине этих движений у шахматистов различной квалификации. Человек, почти не умеющий играть в шахматы, осматривает доску почти равномерно, фиксируя глаза почти с равными интервалами по всему полю доски. Шахматисты высокой квалификации не делают этого. После одного общего обзорного движения по доске они начинают фиксировать глаза только на наиболее опасных и перспективных местах, многократно возвращаются к ним, связывая свои фигуры и фигуры противника, а также свободные поля, относящиеся к планируемым маршрутам перемещения фигур. У шахматных мастеров высшего уровня в движении глаз можно заметить большую глобальность, чем у перворазрядников. Они фиксируют не только «горячие точки шахматной доски», которые имеются в дан-

ной ситуации или возникнут через два-три полухода, а просматривают развитие партии на большее число полуходов, что заставляет их совершать более длинные перемещения глаз. Ведущийся одновременно с киноре¬гистрацией протокол «рассуждения вслух» подтверждает это положение.

Для того чтобы сформировать лабиринт не слитком большого размера, игроки в «Игру в 5» и испытуемые шахматисты стараются укрупнить элементы, из которых складываются ситуации. Это видно из того, что точки фиксации связывают между собой такие, например, пары шахматных фигур, как сдвоенные пешки, или поля, образующие вилку. Человек как бы связывает между собой отдельные элементы в ситуациях, устанавливает между ними определенные связи и отношения, которые формируют модель проблемной ситуации. Этот процесс В. Н. Пушкин назвал *структуризацией.* Он высказал гипотезу, что именно структуризация лежит в основе той метапроцедуры, которая была названа нами процедурой построения перспективного лабиринта.

**Структуризация.** Итак, В. Н. Пушкин показал, что в основе метапроцедуры формирования нового лабиринта возможностей лежит еще одна метапроцедура — структуризация описания проблемней ситуации. Суть ее состоит в вычленении в проблемной ситуации некоторых базовых элементов и установлении между ними связей, выражаемых отношениями между этими элементами. Эти отношения могут быть бинарными, связывающими два элемента структуры, тернарными, объединяющими в некоторую подструктуру три элемента, или отношениями большей размерности.

Вернемся к «Игре в 5». Элементами ситуации в этой игре выступают клетки поля, перенумерованные цифрами 1, 2, …. 6 (будем обозначать их буквами $a_i$), в фишки, перенумерованные цифрами 1,2, …. 5 (их мы будем обозначать буквами $b_i$). В качестве отношений введем следующие бинарные отношения: «находиться на», «быть слева», «быть справа», «быть снизу», «быть сверху». Эти отношения будем соответственно обозначать через $r_1$, $r_2$, $r_3$, $r_4$ и $r_5$. Тогда любая ситуация, фиксируемая на игровом поле, может быть структурирована с помощью выделенных элементов и отношений. Например, исходная ситуация, заданная



в виде


может быть описана следующей структурой:

$(b_1r_1a_1)$, $(b_3r_1a_2)$, $(b_5r_1a_3)$, $(b_2r_1a_4)$, $(b_4r_1a_6)$, $(b_1r_2b_3)$, $(b_1r_5b_2)$, $(b_3r_3b_1)$, $(b_3r_2b_5)$, $(b_3r_5a_5)$, $(b_5r_3b_3)$, $(b_5r_5b_4)$, $(b_2r_4b_1)$, $(b_2r_2a_5)$, $(a_5r_3b_2)$, $(a_5r_4b_3)$, $(a_5r_2b_4)$, $(b_4r_3a_5)$, $(b_4r_4b_5)$.

Конечно, полученная структура описывает ситуацию избыточным образом, ибо отношения «быть слева» и «быть справа», «быть снизу» и «быть сверху» не являются независимыми. Но нас пока но интересует проблема минимизации структуры. Если такое же описание получить и относительно второй целевой ситуации, то можно формальным обрезом вычленить те различия, которые имеются между расположением фишек в первой и второй позициях. Они будут определяться всеми несовпадающими в структурном описании тройками. Если, например, целевая ситуация имеет вид


то различие в структурах исходной и целевой ситуаций коснется лишь фишек с номерами 3, 4 и 5 и свободного поля. Само различие может быть задано перечислением отличающихся пар:

| исходная позиция | целевая позиция |
|---|---|
| $(b_3r_2b_5)$ | $(b_3r_2b_4)$ |
| $(a_5r_2b_4)$ | $(a_5r_2b_5)$ |
| $(b_4r_3a_5)$ | $(b_4r_3b_3)$ |
| $(b_4r_4b_5)$ | $(b_4r_4b_4)$ |
| $(b_5r_5b_4)$ | $(b_4r_5b_5)$ |
| $(b_5r_3b_3)$ | $(b_5r_3a_5)$. |