**М.В. АРЖАКОВ**
**Н.В. АРЖАКОВА**
**В.К. ГОЛИКОВ**
**Б.Е. ДЁМИН**
**В.И. НОВОСЕЛЬЦЕВ**

# МОДЕЛИРОВАНИЕ СИСТЕМ

**Издательство «Научная книга»**
**Воронеж 2005**

ББК 66.4(0)
УДК 316.285
**А 80**

**А 80** **Аржаков М.В., Аржакова Н.В., Голиков В.К., Дёмин Б.Е., Новосельцев В.И.** МОДЕЛИРОВАНИЕ СИСТЕМ / Под ред. В.И. Новосельцева.– Воронеж: Изд-во «Научная книга», 2005. – 216 с. – ISBN

Монография посвящена систематизированному изложению теоретических основ моделирования систем. Включает четыре части: общие положения, теория логико-лингвистического моделирования, модели оптимизации и нейросетевое моделирование

Прочитав книгу, Вы получите ответы на следующие вопросы. Что такое модель системы? Как моделировать системы и какова схема модельных исследований? Что представляют собой «мягкие вычисления», нечеткие множества, семантические сети и логико-лингвистические модели? Чем занимается наука нейроинформатика и как строить модели с помощью искусственных нейронных сетей? В чем суть генетических алгоритмов и можно ли их использовать для поиска оптимальных решений?

Предназначена для научных работников и специалистов в области управления социальными и экономическими системами. Будет полезна аспирантам и студентам старших курсов, специализирующихся на проблемах математического моделирования и информатизации экономических, технологических и других процессов.

ББК 66.4(0)
УДК 316.285

На первой странице обложки: картина воронежского художника М.Д. Викторова «Песочные часы»



**ISBN**

# ОГЛАВЛЕНИЕ

# ПРЕДИСЛОВИЕ

В современной науке моделирование рассматривается как основной и наиболее перспективный метод научного познания действительности, связанный с совершенствованием способов получения и фиксации информации об изучаемых объектах, а также с приобретением новых знаний на основе модельных экспериментов. Особенно возросла роль моделирования с развитием компьютерных информационных технологий.

Исторически моделированию предшествовал метод научного эмпиризма (от греч. *empeiria* – опыт). Эмпирические исследования ограничиваются наблюдениями, сбором информации, классификацией изучаемых явлений и формулированием выводов на основе логических умозаключений. Эмпиризм и сегодня не потерял своего научного значения как метод, предваряющий и сопровождающий любое модельное познание. Вместе с тем, доминирование эмпиризма свидетельствует о застое в данном научном направлении, отсутствии новых конструктивных идей методологического плана, а зачастую и о консерватизме работающих в нем ученых и специалистов.

Эмпиризм, с его буквальным пониманием известного лозунга «практика – критерий истины», породил один из самых порочных методов исследования систем, так называемы метод «проб и ошибок», суть которого выражается фразой: давайте сделаем нечто, затем посмотрим, что получится, и если выйдет плохо, то подкорректируем это нечто. Прикрываясь статистикой, тестированием, опросами общественного мнения, логикой, историзмом и другими схемами этот метод получил тотальное распространение как в естественнонаучных, так и в гуманитарных областях исследования. В первом случае его применение чревато неоправданно большими временными и экономическими затратами, во втором – этот метод приносит страдания и горести тем, на ком практическим путем устанавливается «истина».

Философской базой моделирования выступает теория отражения, точнее, исходный постулат об отражении как специфическом взаимодействии двух систем, в результате которого одна система воспроизводится в другой. В научных исследованиях свойство отражения получает форму взаимодействия реальности и человеческого сознания. Реальность через органы чувств воспринимается человеком и воздействует на его сознание, в результате чего формируется некий слепок – образец той реальности, которую мы наблюдаем. Этот слепок, называемый моделью (от фр. *modèle* – образец), воспроизводится каким-либо способом на том или ином носителе информации и служит объектом исследования.

Таким образом, модель несет в себе информацию о реальности, воспринятую субъектом и выраженную им в форме мыслительной конструкции, рисунка, математической формулы, словесного текста, графического изображения, компьютерной программы и т.д. Следовательно, для одного и того же реального объекта или процесса можно построить совершенно разные модели, отражающие индивидуальный взгляд того или иного исследователя (или группы исследователей) на объект изучения. Поэтому утверждается, что любая модель (независимо от способа ее выражения) субъективна по своему содержанию.

Именно этот тезис долгое время служил камнем преткновения на пути признания моделирования эффективным инструментом научных исследований. Философские дискуссии о том, что считать объективным, а что – субъективным, не затихают и по сей день. С одной стороны, все объекты реального мира мы воспринимаем через призму собственного понимания их сути, и в этом смысле все то, что мы видим, слышим, чувствуем, с чем имеем дело, не более чем наше субъективное представление о происходящем. С другой стороны, в потоке информации, поступающей в сознание человека через органы чувств, содержатся вполне определенные данные о наблюдаемых объектах, не зависящие от вос-

принимающего их субъекта. В этом аспекте можно говорить об объективном характере восприятия действительности.

В теории моделирования в качестве основного постулата принято положение о том, что сама природа восприятия действительности имеет двойственную объективно-субъективную сущность. Причем эти стороны не исключают, а диалектически дополняют друг друга. В результате образуются своеобразные информационные конфликты (споры, разногласия, сомнения, противоречия), выступающие движущей силой никогда не завершающегося процесса познания и установления истины.

Согласно этому положению, изучая какой-либо объект методом моделирования, исследователь буквально обречен на бесконечный циклический процесс «модель → апробация → корректировка → уточненная модель…», констатируя всякий раз лишь определенный уровень познания и относительный характер истинности добытых сведений. Поэтому, оценивая адекватность модели, можно говорить лишь о степени ее приближения к объекту-оригиналу, понимая, что точного соответствия не может быть в принципе.

Этим утверждением определяется критерий адекватности, принятый в теории моделирования: пригодность модели конструктивно разрешать конкретные проблемы. Другими словами, модель считается адекватной реальности, если выражаемые ею закономерности не противоречат наблюдаемым фактам, а получаемые с ее использованием выводы позволяют достичь целей данного исследования.

В своем историческом развитии метод моделирования прошел три основных этапа. Причем каждый последующий этап не отрицал, а дополнял и расширял предыдущий, включая в себя все то лучшее и конструктивное, что было достигнуто предшествующими исследователями.

Для ***первого (физикалистического) этапа*** характерно стремление построить модели не только простых, но и сложных

систем на основе известных и еще не открытых законов физики. Утвердившись в науке с середины XVII века, такой подход оказался чрезвычайно плодотворным при изучении вещественно-энергетических преобразований, но обнаружил свою несостоятельность при попытках познания структурно-поведенческих сторон процессов и явлений. Вера в простоту устройства нашего мира ушла ныне в невозвратное прошлое. Все объекты окружающей нас действительности демонстрируют системный многоуровневый принцип своей организации. С пониманием этого стала очевидной иллюзорность попыток объяснить мироустройство с помощью простых и изящных моделей типа гамильтоновых уравнений классической механики, волновых уравнений квантовой механики или уравнений электромагнитной динамики.

***Второй (операционный) этап*** характеризуется прорывом специальных математических методов в сферу модельных исследований. Методологическую основу моделей этого периода составила теория исследования операций со всеми ее многочисленными разделами: линейным, нелинейным, динамическим программированием, игровыми моделями, методом Монте-Карло, структурным моделированием, агегативным подходом и др.

Как известно, в операционных исследованиях разрешение проблемы достигается путем ее идеализации до уровня, позволяющего выразить сущность на математическом языке, то есть разработать математическую модель явления, задать в количественном виде критерии выбора решений и установить ограничения на варьируемые параметры. Наличие количественных критериев и формальных моделей позволяет сформулировать проблему в терминах математической оптимизации и свести ее решение к поиску алгоритма, позволяющего найти за конечное число шагов наилучший вариант относительно заданных критериев при фиксированных ограничениях. Иными словами, в рамках операционного подхода проблема считается разрешимой, если она трансформируема в оптимизационную задачу, и эта задача может быть реше-

на на основе известных методов математического программирования или их модификаций. Такой путь хоть и изящен с математической точки зрения, но фактически означает подгонку проблемы под возможности метода, то есть предполагает доминирование метода над существом проблемы. С системной позиции этот подход не может быть признан конструктивным, так как получаемые при этом выводы и рекомендации справедливы только по отношению к созданной математической модели и приемлемы только тогда, когда данная модель является исчерпывающим представлением практической проблемы, что далеко не всегда соответствует действительности.

Практические проблемные ситуации характерны тем, что в них не только не представляется возможным корректно определить понятие оптимальности, но даже на вербальном уровне задать достаточно полную модель явления. По существу, для любой системной проблемы свойственно отсутствие какой-либо модели, устанавливающей исчерпывающим образом причинно-следственные связи между ее компонентами, а о существовании критериев оптимальности можно говорить только после разрешения проблемы.

Условность оптимального варианта разрешения сколько-нибудь значимой практической проблемы – факт общепризнанный. Достаточно назвать вариант, претендующий на эту роль, как не составит большего труда найти ряд обстоятельств, которые не были учтены при его обосновании, и тем самым продемонстрировать условность оптимальности. То есть, сделать вывод о том, что данный вариант можно признать оптимальным при условии, если … и далее следует перечень ограничений и допущений, позволивших свести реальную проблему к оптимизационной математической задаче. Конечно, можно модифицировать метод и снять ряд ограничений и допущений, но тогда вскроются новые неучтенные обстоятельства, и такой процесс может повторяться неограниченно долго, всякий раз констатируя условную оптималь-

ность. Условная оптимальность приемлема в теории, но не на практике, где она проявляется в виде ошибочных решений и неверных действий.

Традиционно считалось, что все неудачи операционного подхода к разрешению практических проблем связаны с недостаточным развитием математических методов оптимизации или обусловлены неадекватностью математической модели объекту исследования. Но оказывается, что дело не в математике и не в способах моделирования, а в принципе: в человеческой деятельности не существует оптимальных (абсолютно верных) решений – так же, как не бывает неразрешимых проблем (абсолютно тупиковых ситуаций).

На смену господствовавшему принципу экстремальности приходит компромиссный принцип разрешения системных проблем, когда оптимальность рассматривается в ее широком диалектическом смысле – как никогда не прекращающийся процесс поиска компромисса между потребностями, возникающими в результате развития индивида и общества, и возможностями их удовлетворения на базе формирования новых гуманитарных, промышленных, экономико-финансовых и других технологий.

***Третьему (системному) этапу*** свойственно то, что объект моделирования рассматривается как система, а задача заключается в комплексном и всестороннем изучении ее поведенческих аспектов с учетом внешнего окружения. Методологическую основу построения моделей этого периода составил системный подход совместно с технологиями системного анализа. При этом постулируется, что изучаемый объект представлен в данном исследовании как система, если независимо от его субстанциональной сущности и физических размеров он идентифицируется по признакам разделяемости, целостности, связанности и неаддитивности, а само исследование относится к классу системных, если процедурно оно строится без нарушения положений этих признаков [Новосельцев, 2003]. Соответственно центр тяжести модельных иссле-

дований переместился от изучения свойств отдельных компонентов системы к исследованию связей между ними, и выявлению на этой основе эмерджентных качеств изучаемого объекта.

Такая точка зрения потребовала коренного пересмотра самого подхода к моделированию изучаемых объектов, а также послужила толчком к развитию специальных методов моделирования, адекватных по своим познавательным возможностям уровню сложности моделируемых объектов.

Моделирование систем (или системное моделирование), ставшее возможным благодаря прорывному развитию компьютерных информационных технологий, – это ответ прикладной науки на вызов, который был ей брошен сложными системами.

Системные модели не следует рассматривать как антитезу традиционному математическому моделированию. Здесь имеет место симбиоз и содружество, когда стирается грань между стремлением к тотальной формализации и логико-интуитивным (эвристическим) подходом к анализу системных явлений. Таким образом, системное моделирование есть разумный компромисс между этими крайними точками зрения на возможные пути конструктивного разрешения сложных проблем, для которых характерны следующие черты: слабая структурированность, конфликтность, неопределенность, неоднозначность, наличие риска, многоаспектность, комплексность, саморазрешимость, эволюционность.

Системное моделирование – сравнительно молодое и далеко незавершенное научное направление, в котором пока больше проблем и нерешенных вопросов, чем успехов и достижений. В настоящее время развитие этого направления существенно сдерживается отсутствием конструктивной технологии проектирования системных моделей. Концепция такой технологии очевидна: системная модель – это самоорганизующаяся многоуровневая система, следовательно, проектировать ее надо так, как проектируются любые другие системы такого класса. Однако, также оче-

видны трудности реализации такой концепции – пока нет универсальной и достаточно формализованной технологии проектирования самоорганизующихся символьных систем, инвариантной к предметным областям.

На практике применяется достаточно много схем, претендующих на роль такой технологии. Но все они эвристичны по своей сути, чрезмерно обобщены и отражают не более чем опыт тех или иных исследователей. Поэтому при овладении технологией проектирования системных моделей целесообразно опираться на методы и стандарты обеспечения качества программных средств [Липаев, 2001], соблюдение требований которые следует рассматривать как необходимое условие создания эффективных исследовательских моделей.

Важнейшим отличием системного моделирования от традиционного является использование языков представления знаний, позволяющих отражать различные аспекты проблемной области, наблюдаемые факты и закономерности, описывать эти знания не только на количественном, но и на качественном уровне. По сути, системное моделирование – это процесс восприятия, фиксации, переработки и получения новых знаний на базе использования информационных компьютерных технологий. Поэтому проблема представления знаний (подчеркнем еще раз – не данных, а знаний) находится в центре внимания специалистов по системному моделированию. Проблема включает в себя множество трудных аспектов, но кардинальным является вопрос о механизмах представления знаний и структуре самого языка этого представления. Исчерпывающего ответа на него пока нет. Частично ответить на этот вопрос удалось при изучении способов построения логико-лингвистических и нейросетевых моделей, а также генетических алгоритмов.

# ГЛАВА 1. ОБЩИЕ ПОЛОЖЕНИЯ

## 1.1. Понятие модели

Сущность метода моделирования состоит в том, что наряду с системой-оригиналом, которую мы обозначим через $S^0$, рассматривается ее модель, в качестве которой выступает некоторая другая система S, представляющая собой образ (подобие) оригинала $S^0$ при моделирующем отображении (соответствии подобия), что принято обозначать записью: $f: (S^0) \rightarrow S$, где скобки означают, что f – частично определенное отображение (то есть, не все черты оригинала отражаются моделью).

Моделирующее отображение f обычно представляют в виде композиции (продукта последовательного выполнения) двух отображений – огрубляющего g и гомоморфного h (от греч. *homos* – одинаковый + греч. *morphē* – форма): $g: (S^0) \rightarrow S^1$; $h: S^1 \rightarrow S$; $f = h \circ g: (S^0) \rightarrow S$, где $S^1$ некоторая подсистема системы $S^0$.

Модель, как правило, представляет собой упрощенный образ оригинала, и это упрощение (огрубление) осуществляется отображением g , при котором, сознательно удаляя из системы $S^0$ некоторые компоненты и связи, мы получаем подсистему $S^1$. В то же время модель должна в определенном смысле верно отражать оригинал, хотя, возможно, и огрублено, или агрегировано. Именно это и осуществляет гомоморфное отображение h подсистемы $S^1$ на модель S.

В зависимости от характера огрубления и степени агрегирования для одного и того же оригинала можно получить несколько различных моделей. Стратегия моделирования заключается в попытке путем упрощения получить модель, свойства и поведение которой можно было бы эффективно изучать, но которая в то же время оставалась бы сходной с оригиналом, чтобы результаты изучения были к нему применимы.

Обратный переход от модели S к оригиналу $S^0$ называется интерпретацией модели. Процедура интерпретации не является строго однозначной, так как прообраз некоторых компонентов или отношений модели в силу необратимости гомоморфного отображения h может состоять из нескольких компонентов или отношений системы-оригинала.

Оригинал и модель, а также разные модели одного и того же оригинала могут отличаться по своей реализации, где под реализацией понимается способ моделирующего отображения. В зависимости от особенностей системы-оригинала и задач исследования применяются самые различные способы моделирующего отображения, а сами модели получают соответствующую классификацию (рис. 1.2).


Рис. 1.2. *Классификация моделей (вариант)*

В современных научных исследованиях используются как реальные (натурные, аналоговые), так и знаковые (идеальные) модели, однако в связи с широким развитием компьютерных технологий наибольшую значимость приобретают знаковые модели. В качестве других причин, определяющих их преимущественное использование можно указать:

- недопустимость или нежелательность стороннего вмешательства в функционирование изучаемого объекта, в частности потому, что оно может нарушить, либо исказить естественное развитие процесса;
- техническую и технологическую сложность постановки и проведения натурных экспериментов, а также недопустимо высокие затраты, потребные для их организации и проведения;

- недоступность или опасность натурного изучения объекта, например, из-за его значительной удаленности, или вследствие активного противодействия со стороны противника;
- отсутствие изучаемого объекта в действительности, что, в частности, характерно для проектирования новых систем.

Помимо этого, знаковые модели служат своего рода банком знаний, хранящим в себе сведения о составе, структуре и поведении изучаемых систем, имеющих зачастую непреходящую научную и практическую ценность.

Знаковые, или идеальные, модели представляют собой условное описание системы-оригинала с использованием заданного алфавита символов и операций над символами, в результате чего получаются слова и предложения некоторого языка, которые с помощью определенного кода интерпретируются как образы компонентов системы-оригинала и взаимодействий между ними. В зависимости от используемых языков знаковые модели бывают описательными (вербальными), формальными и формализованными (или логико-лингвистическими).

Описательные (вербальные) модели разрабатываются на основе естественных языковых средств. Как правило, они состоят из научных текстов, сопровождаемых блок-схемами, таблицами, графиками и прочим иллюстративным материалом. Их основное назначение – служить обобщенным и в то же время достаточно полным выражением знаний исследователя об изучаемой системе в пределах средств определенной научной концепции. Эти модели неразрывно связаны с формальными и формализованными.

С одной стороны, в них содержатся исходные данные, необходимые для построения формальных и формализованных моделей, а с другой – они выступают наглядной формой выражения результатов исследований, представляемых заказчику. Важность последнего аспекта часто недооценивается. Это приводит к тому, что многие чрезвычайно важные научные результаты не воспри-

нимаются лицами, ответственными за принятие решения, и долгое время остаются невостребованными.

Формальные модели представляют собой способ концентрированного выражения знаний, представлений и гипотез о системе-оригинале в виде математических соотношений. Они используются для описания хорошо структурированных проблем, где свойства изучаемых объектов и соотношения между ними можно выразить в количественной форме. Для построения таких моделей привлекается все многообразие средств современного математического аппарата – алгебры, геометрии, теории дифференциального и интегрального исчисления, теории вероятностей и математической статистики, теории оптимального управления и т. д. Особая значимость формальных моделей определяется тем, что они позволяют оценивать эффективность систем по количественным показателям.

Оценка эффективности изучаемой системы сводится к установлению функциональных зависимостей вида:

$$Э(t) = F[X(t), Y_x(t), Z(t)]|_{V(t)}, \qquad (1.1)$$

где Э(t) – множество количественных показателей эффективности системы, X(t) – множество компонентов системы (ее состав), $Y_x(t)$ – множество характеристик компонентов системы, Z(t) – множество отношений, связей и взаимодействий компонентов системы между собой (структура системы), V(t) – множество характеристик среды, влияющих на функционирование системы, t – время.

Процесс получения таких зависимостей называется математическим моделированием.

В зависимости от свойств функционала F математические модели классифицируются по разным признакам. Так, если для F найдено точное математическое выражение, позволяющее для любых входных переменных X(t), $Y_x(t)$, Z(t) и заданных внешних условий V(t) непосредственно определять значение показателей эффективности Э(t) в любой нужный момент времени t, то модель

принято называть аналитической. Аналитические модели обладают многими качествами, облегчающими их исследование и применение. Однако в подавляющем большинстве случаев нахождение аналитического выражения для F оказывается затруднительным, либо в принципе невозможным. На практике чаще всего функционал F удается задать в виде компьютерного алгоритма (программы), с помощью которого рассчитываются значения показателей эффективности на интервале времени $t_0 \le t \le t_N$. Такие модели называют имитационными (от лат. *imitatio* – подражание).

В зависимости от характера связи между Э(t) и X(t), $Y_x(t)$, Z(t), V(t) математические модели бывают детерминированными и стохастическими. Если в детерминированной модели показатели эффективности определяются однозначно (с точностью до ошибок вычисления), то стохастическая модель дает для каждого показателя распределение возможных значений, характеризуемое математическим ожиданием, среднеквадратическим отклонением и другими моментами.

По характеру временного описания моделируемого объекта различают дискретные и непрерывные модели. Дискретная модель описывает поведение системы на фиксированной последовательности моментов времени $t_0 < t_1 < \ldots < t_j < \ldots < t_N$, тогда как в непрерывной модели значения показателей эффективности могут быть рассчитаны для любой точки t рассматриваемого интервала $[t_0, t_N]$. Среди дискретных выделяются модели с фиксированным шагом по времени ($\Delta t = t_j - t_{j-1} = \text{const}$ для всех j от 0 до N), который не зависит от результатов моделирования и не может быть изменен без глубокой перестройки всей модели. Такими моделями адекватно описываются системы, поведение которых не связывается с внутренним временем, или время считается однородным. Существуют дискретные модели с переменным временным шагом, который уменьшается или увеличивается в зависимости от результатов моделирования

$$\Delta t = T[Э(t)], \qquad (1.2)$$

где T – оператор внутреннего времени.

В частности, такие модели используются для описания функционирования многоуровневых самоорганизующихся систем, в которых внутреннее время имеет неоднородную иерархическую структуру.

Следующий признак, по которому различаются математические модели, – это характер описания пространственного строения объекта моделирования. Модели, в которых пространственное строение системы не учитывается, принято называть моделями с сосредоточенными параметрами (или точечными моделями), в отличие от моделей с распределенными параметрами, в которых показатели эффективности зависят не только от времени, но и от положения моделируемого объекта в некоторой системе координат.

Формализованные модели служат для описания слабо структурированных проблем. Это тоже математические модели, но строятся они на основе иных языковых средств (нечетких множеств, реляционных, фреймовых языков) и реализуются на компьютерах в виде логико-лингвистических моделей.

Завершая краткий обзор модельной типологии, отметим, что рассмотренная классификация моделей весьма условна и в определенной мере препятствует целостному восприятию изучаемых объектов, поскольку отражает фактически бессистемный подход к моделированию систем. Поэтому изложенное есть не более чем экскурс в историю моделирования. Конечно, историю необходимо знать, поскольку без знания минувшего нет настоящего и будущего. Однако не следует уповать на прошлое в такой быстро развивающейся области, как моделирование систем. За последние десятилетия, благодаря развитию компьютерных технологий, значительно расширились возможности по имитации изучаемых объектов и, соответственно, усовершенствовались методы построения моделей систем. В теории моделирования были выдвинуты новые принципы, практическое воплощение которых позво-

лило сформировать новое научное направление, получившее название системного моделирования.

## 1.2. Основная концепция системного моделирования

Главное требование к любой модели состоит в том, чтобы она была адекватна объекту изучения, иначе теряется смысл моделирования. Очевидно, что создание адекватной модели возможно только в том случае, когда свойства и взаимосвязи моделируемого объекта известны и в достаточной степени изучены. Но если объект изучен, зачем его моделировать? И наоборот, если объект не изучен, тогда как можно построить его адекватную модель? Налицо парадокс, имеющий место не только в системных, но и в любых других исследованиях.

В традиционных научных направлениях он разрешается тем, что модель не обосновывается, а постулируется на основе тех немногих эмпирический сведений, которыми располагает исследователь на текущий момент времени. Так, например, в классической и квантовой механике второй закон И. Ньютона (основная модель механики макромира) и волновое уравнение Э. Шредингера (основная модель микромира) не выводятся из каких-либо предпосылок, а постулируются. Уравнения Д. Максвелла, описывающие динамику электромагнетизма, также не доказываются, а принимаются как аксиомы. Такой же подход прослеживается в теоретической биологии, где логистическое уравнение, с помощью которого описывают динамику биологических популяций, принимается как исходное и не доказывается.

В период своего становления системное моделирование развивалось примерно по такому же пути. Из математики заимствовался какой-либо подходящий метод, который модифицировался и дорабатывался с учетом особенностей системы-оригинала, насыщался соответствующей терминологией, доводился до вычислительных процедур и представлялся как модель системы. Затем проводились исследования этой модели, по результатам которых

формулировались выводы и выдавались рекомендации по рациональным способам его поведения в тех или иных ситуациях.

При этом в неявном виде постулировалось, что аксиоматика, принятая при разработке математического метода, соответствует принципам построения и существу функционирования того реального объекта, для моделирования которого использовался данный метод. Так, например, считалось, что методы теории массового обслуживания одинаково пригодны для имитации процессов функционирования систем связи и процессов ведения боевых действий. Такую концепцию построения системных моделей можно назвать редукционизмом (от лат. *reductio* – возвращение, приведение обратно, сведение сложного к простому).

Развитие компьютерных технологий на первых порах вселило надежду, что, облекая системные проблемы в математические формы, можно наконец-то найти в сфере математики ключ к пониманию универсальных законов развития систем. Однако реальность оказалась значительно сложнее, и первоначальная эйфория уступила место разочарованию. Выяснилось, что полная формализация процессов, происходящих в природе и обществе, невозможна, и что законы математики не являются абсолютной истиной. Исчезла уверенность, что математические методы обладают внутренним содержанием, одинаково пригодным для интерпретации различных по своей природе сущностей.

В настоящее время принята иная концепция моделирования систем, получившая название гомеостатической (от греч. *homoios* – подобный + *status* – состояние). На практике она реализуется различными способами, но суть у них одна: пошаговое приведение исходной модели к состоянию, подобному объекту-оригиналу, за счет включения в модель программных механизмов адаптации и интерпретации, а также организации режима эффективного диалога с исследователем.

Идея построения гомеостатической модели проста, но ее практическое воплощение требует привлечения принципиально

новых информационных технологий. На первом шаге, используя данные описательной модели, строится так называемый каркас системной модели (ее исходное, нулевое приближение), учитывающий априори известные свойства и аспекты моделируемой системы. Этот каркас далек от адекватности объекту-оригиналу и не позволяет сформулировать сколько-нибудь значимые практические выводы, но одновременно в него закладываются специальные алгоритмы, позволяющие изменять исходные предпосылки (базовые аксиомы и правила вывода) по мере получения новых данных об объекте изучения. Далее проводится модельный эксперимент. Полученные при этом данные используются для корректировки каркаса – формируется модель системы в первом ее приближении. Затем уже с помощью этой модели проводится эксперимент, по результатам которого она вновь корректируется – формируется модель системы во втором ее приближении, и так далее. Такой циклический обучающий процесс «эксперимент – данные – корректировка» многократно повторяется и никогда не завершается построением окончательной системной модели. Всегда это будет некое приближение к системе-оригиналу, нуждающееся в уточнении в ходе дальнейших исследований. Адекватность системной модели объекту изучения нельзя доказать – она может быть либо принята как временное соглашение, либо отвергнута на том основании, что получаемые с ее помощью оценки и выводы противоречат наблюдаемым фактам и не позволяют достичь целей исследования. Системная модель всегда будет отличаться от оригинала и может лишь асимптотически приближаться к нему при выполнении определенных условий, специфичных для каждой практической задачи.

Гомеостатическая концепция моделирования не гарантирует сама по себе сходимости модели и изучаемого объекта. На практике асимптотическая сходимость «модель-объект» обеспечивается тем, что объектом моделирования выступает конкретная система, с присущими только ей автономными законами функциони-

рования. Автономные законы не распространяются на системы вообще, они свойственны и присущи только данной системе. Принципиальным здесь является то, что адекватность достигается сужением сферы использования данной системной модели, ограниченностью ее практической применимости. В пределе каждая системная модель уникальна в той же степени, в какой уникальна каждая система-оригинал.*) Кроме того, адекватность системной модели может быть повышена за счет использования результатов натурных и лабораторных экспериментов (пусть отрывочных и неполных). У исследователя в ряде случаев существует возможность сопоставить теорию с практикой и внести в модель соответствующие поправки (другой вопрос – во что это выливается, и что считать более правильным – наблюдаемое или предсказываемое теорией).

История науки свидетельствует о том, что далеко не всегда явно наблюдаемый или ненаблюдаемый факт есть истина. Хрестоматийным примером в этом отношении может служить открытие планеты Нептун, когда прямые астрономические наблюдения не позволяли зафиксировать ее присутствие в Солнечной системе, а теоретические расчеты говорили о том, что такая планета должна существовать. Теоретики оказались правыми. Точно в указанный момент времени и доподлинно в предсказанном месте берлинским астрономом Галле в 1846 году была действительно обнаружена неизвестная планета Солнечной системы, позже названная Нептуном.

Наконец, адекватность системной модели повышается за счет самих модельных экспериментов. Модельные эксперименты стимулируют появление новых знаний интуитивного свойства, кото-

*) Разработанную и апробированную на практике системную модель, разумеется, можно и нужно использовать для разрешения разнообразных проблем. Но при этом во главу угла должны ставиться специфические особенности этих проблем и объектов их изучения, а не вычислительные и логические возможности, заложенные в модель. Иначе модель будет доминировать над существом дела, а моделирование превратится в самоцель.

рые используются для самонастройки модели и приближения ее свойств к свойствам изучаемого объекта. Механизм этого явления пока не вскрыт, но факт остается фактом: сам процесс моделирования позволяет исследователю более глубоко проникнуть в существо объекта-оригинала, а модельные исследования приводят к открытию новых свойств и закономерностей функционирования изучаемой системы даже в том случае, когда модель не соответствует оригиналу.

Принцип построения системной гомеостатической модели иллюстрируется схемой на рис. 1.3. Помимо обычной формальной системы (Ф), в ее состав включаются две подсистемы – адаптации и интерпретации, которые во взаимодействии с исследователем и реализуют модельный гомеостаз. Формальная система задается четверкой:

$$\Phi = <T, C, A, П>, \qquad (1.3)$$

где Т – термы (алфавит) формальной системы, то есть базовые понятия и символы, используемые для конструирования формул; С – синтаксис, то есть правила построения правильных формул; А – аксиомы, то есть правильно построенные формулы, выражающие утверждения, которые считаются истинными априори; П – правила вывода новых формул, позволяющие выводить из аксиом новые утверждения.

Формальная система, входящая в состав системной гомеостатической модели, имеет существенные отличия от обычных формальных конструкций.

Во-первых, в ней предусматривается возможность оперативного изменения аксиом и правил вывода. Этими изменениями управляют подсистемы адаптации и интерпретации. Подсистема адаптации конструирует новые правила вывода и вносит соответствующие изменения в формальную систему. Подсистема интерпретации изменяет ее аксиоматику, то есть вводит в формальную

систему новые аксиомы и удаляет старые. Всеми операциями ввода-вывода управляет исследователь.

Во-вторых, формальная система – это не формула и не совокупность математических уравнений. Она включает в себя самые разнообразные математические, логические и иные модули, которые необходимы для решения поставленной проблемы и из которых по определенным правилам конструируются различные алгоритмы решения исследовательских задач. В качестве таких модулей могут использоваться, например, стандартные программы решения систем дифференциальных уравнений определенного типа, программы поиска критического пути на графах, программы решения задач линейного программирования симплекс-методом, программы приведения логических высказываний к конъюнктивной (дизъюнктивной) нормальной форме и другие.

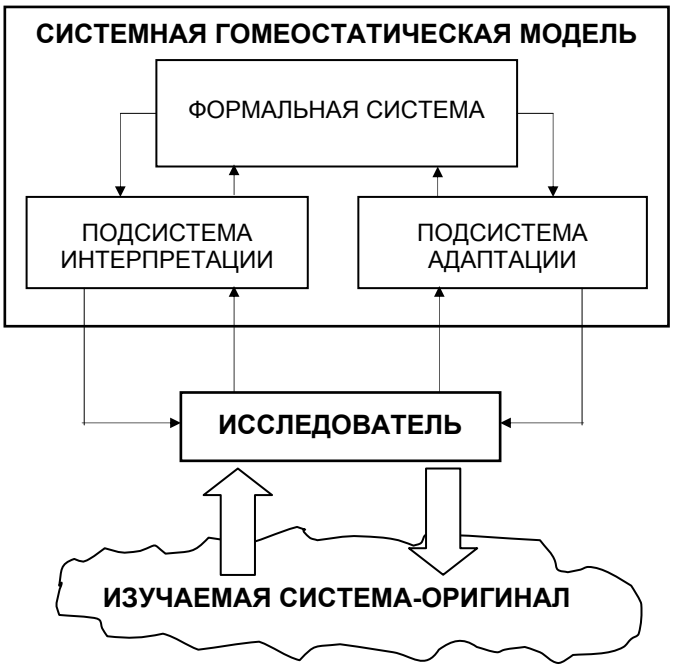

Рис. 1.3. *Принцип построения системной гомеостатической модели*

Помимо чисто математических модулей, в состав формальной системы включаются проблемно-ориентированные методики, алгоритмы и программы, позволяющие рассчитывать показатели эффективности, характерные для конкретной проблемной области. Так, например, если объектом изучения выступает некая производственно-экономическая система, то в состав формальной системы должны входить методики расчета таких показателей, как прибыль, рентабельность, окупаемость капиталовложений, себестоимость продукции, налоговые отчисления, уровень запасов по видам изделий и т. д.

В третьих, в отличие от обычных формальных моделей, имеющих правила вывода вида $\Pi_i \Rightarrow \Pi_j$ (если $\Pi_i$, то $\Pi_j$), формальная система в гомеостатической модели имеет правила вывода следующей типовой структуры:

$$(\alpha_k \Pi_k)(\Pi_i \overset{O_i}{\Rightarrow} \beta_1 \Pi_1 \vee \beta_2 \Pi_2 \vee ... \vee \beta_j \Pi_j \vee ... \vee \beta_N \Pi_N), \qquad (1.4)$$

где $\Pi_k$ – формула, устанавливающая условие применимости данного правила вывода; $O_i$ – сигнал, свидетельствующий о реализации данного правила вывода; $\alpha_k$ $\beta_j$ – кванторы, имеющие смысл вероятности или принимающие такие лингвистические значения, как «часто», «редко», «иногда», «почти всегда» и другие.

Указанное правило вывода в том случае, если $\alpha$ принимает значение «в большинстве случаев», $\beta$ – «иногда», а $N = 1$, читается так: в большинстве случаев, если утверждение $\Pi_k$ справедливо, то из посылки $\Pi_i$ иногда следует заключение $\Pi_j$ и при этом вырабатывается сигнал $O_i$. Этот сигнал для самой формальной системы не нужен, он обеспечивают обратную связь этой системы с подсистемами адаптации и интерпретации, информируя их о том, что произошла реализация данного вывода. Истинность формулы $\Pi_k$ устанавливается двумя способами: либо $\Pi_k$ получается в результате вывода в формальной системе из имеющихся аксиом, либо ее истинность предписывается подсистемой интерпретации. Истинность $\Pi_k$ в подсистеме интерпретации устанавливается также двумя способами: либо она получает от исследователя утверждение об истинности $\Pi_k$, либо производится логический вывод из хранящихся в подсистеме интерпретации фактов. В первом случае реализуется режим обучения модели, во втором – режим модельного эксперимента.

Несмотря на внешнюю простоту, переход к гомеостатической концепции потребовал коренного пересмотра взглядов на сложившиеся принципы моделирования систем, а также решения ряда чрезвычайно сложных научных проблем. По замыслу создания

системные гомеостатические модели – это открытые человеко-машинные системы, в которых компьютер выступает уже не в качестве быстродействующей логарифмической линейки или удобной пишущей машинки с памятью, а как интеллектуальный партнер системного аналитика, ведущий с ним диалог в реальном масштабе времени. Для ведения эффективного диалога необходимо выполнение следующих условий:

- программный комплекс системной модели должен обеспечивать накопление, формирование, хранение и обработку данных и знаний об изучаемой проблемной области (модель должна «понимать», о чем идет речь в данных исследованиях);
- должна быть обеспечена языковая совместимость программного комплекса модели с исследователем (модель должна адекватно «воспринимать» информацию, сообщаемую человеком, а он должен понимать информацию, выдаваемую моделью);
- по указанию исследователя модель должна уметь планировать необходимые вычисления, производить расчеты, осуществлять логическую обработку сообщаемых ей фактов, а также данных, получаемых в результате вычислений и логических выводов.

Эти достаточно новые и весьма трудные вопросы были успешно решены при создании так называемых диалоговых информационно-логических систем – ДИЛОС [Клыков, 1974; Брябрин, 1983].*) Внешне ДИЛОСы можно сравнить с операционными системами общего программного обеспечения типа Windows. Отличие состоит в том, что операционные системы Windows универсальны, то есть, предназначены для выполнения функций общего пользования, а ДИЛОСы – ориентированы на обеспечение функций интеллектуального свойства, прежде всего – модельного гомеостазиса путем управления работой системной модели.

*) ДИЛОС – аббревиатура и одновременно имя собственное диалоговой системы, разработанной в 70-х годах прошлого столетия в ВЦ АН СССР и предназначенной для решения математических задач безусловной оптимизации, нелинейного программирования и оптимального управления.

Как уже отмечалось, при моделировании систем неизменно возникает необходимость упрощения изучаемого объекта, и, соответственно, общей исследовательской задачи. На практике упрощение достигается путем расчленения (декомпозиции) общей задачи на отдельные частные подзадачи, с последующим их решением и сверткой (композицией) частных результатов. Такой путь вполне приемлем и находит широкое применение в практике анализа систем. Однако без специальных мер он может привести к нарушению целостности, разрушению модельного гомеостазиса и, как следствие, к потере адекватности модели объекту моделирования.

В теории системного моделирования пока нет универсальных методов, позволяющих строго формализовать процедуру декомпозиции и композиции системных проблем. Неоднократно предпринимались попытки использовать для целей исследования сложных систем по частям тензорный анализ Крона [Крон, 1972]. Эти попытки весьма заманчивы, но пока не привели к созданию инженерных методов декомпозиции, которые не нарушали бы целостности представления изучаемых систем. Многие исследователи-практики вполне справедливо считают, что при любом способе декомпозиции системных объектов должна происходить определенная потеря целостности, в связи с чем композиция выливается в необходимость решения задачи координации, то есть поиска компромисса между свойствами целого и свойствами составляющих ее частей.

Вместе с тем, практикой выработан ряд приемов, обеспечивающих возможность декомпозиции проблемы с одновременным сохранением целостности изучаемого объекта. К их числу можно отнести: квантификацию целей, стратификацию объекта моделирования и учет временного фактора.

***Квантификация целей*** предполагает последовательное расчленение общей цели разрешаемой проблемы на взаимосвязанные подцели различных уровней, то есть формирование системной

целевой иерархии. При этом на нижнем уровне иерархии формируется полный неизбыточный набор измеримых целей. Измеримость целей эквивалентна однозначности их определения на каких-либо шкалах (по возможности – количественных). Полнота набора целей предполагает, что достижение целей нижнего уровня позволяет достичь целей верхнего уровня, и в конечном счете – общей цели. Неизбыточность означает, что все цели нижнего уровня направлены на достижение какой-либо цели верхнего уровня. Недопустимы как чрезмерная детализация целей, когда достижение некоторой цели нижнего уровня практически не влияет на достижение целей верхнего уровня, так и недостаточная квантификация, когда цели высших уровней остаются недоопределенными.

При разрешении конкретной системной проблемы существует некий рациональный уровень квантификации целей, который по существу является компромиссом между стремлением к возможно более детальному представлению проблемы и реально существующими возможностями. С построения целевой иерархии начинается решение любой системной проблемы. Эта иерархия (или, как ее иногда называют, дерево целей и задач) служит одновременно неотъемлемым компонентом системной модели, выполняя функцию своеобразного контролера-координатора, следящего за сохранением целостности объекта, который воспроизводится в модели.

***Стратификация*** предполагает условное расчленение объекта моделирования на соподчиненные уровни или страты и его представление в виде комплексной иерархии описаний. Каждую страту можно представить как срез изучаемой системы по горизонтали, проведенный таким образом, чтобы можно было локализовать множества функциональных пространств ее описания, то есть «высветить» определенные грани сущности системы. Тогда, устанавливая отношения соподчиненности между стратами, можно сохранить определенный уровень целостного представления

системы. На возможность выделения отношений соподчиненности указывает выдвинутый в синергетике принцип регулировочных параметров порядка. Смысл этого принципа заключается в том, что в неустойчивых состояниях (состояниях слабой устойчивости) поведение сложной многоуровневой системы как бы упрощается, происходит сжатие управляющих потоков информации между уровнями. Такая компактификация позволяет сократить объем анализируемых межуровневых отношений (связей) до приемлемых размеров без существенных потерь в целостном представлении анализируемой системы [Хакен, 1985].

***Учет временного фактора*** позволяет детализировать общую проблему и в некоторой степени сохранить ее целостность за счет того, что функции и свойства систем, как правило, не проявляются все сразу, одновременно. Во многих случаях существует возможность расчленить динамику системы на непересекающиеся во времени этапы, на каждом из которых проявляется ограниченное число системных функций и свойств. Например, военную операцию можно разбить на ряд следующих друг за другом боевых действий. В свою очередь, боевые действия можно представить в виде последовательно выполняемых задач: разведка, нанесение огневого удара и т.д. Учет фактора времени в сочетании с принципом регулировочных параметров порядка позволяет организовать поэтапное моделирование достаточно сложных системных объектов и отразить в модели реальные свойства исследуемой системы.

* * *

Итак, гомеостатическая концепция системного моделирования, пришедшая на смену редукционизму, основывается на представлении модели как открытой иерархической многоуровневой динамической системы, реализация которой предполагает использование интеллектуальных компьютерных технологий, неотъемлемым компонентом которых является исследователь. Мо-

дели, построенные на основе такой концепции, по своему составу, структуре и функционированию оказываются близкими к объектам-оригиналам с той существенной разницей, что в отличие от реальных систем они могут быть воспроизведены в виде компьютерных алгоритмов и программ – с ними можно производить неограниченное количество модельных экспериментов, безопасных для жизни исследователя. Процесс функционирования таких моделей следует назвать самоорганизующимся, его развитие происходит не по установленной раз и навсегда программе, а определяется результатами, которые получаются в ходе модельного эксперимента. При этом в зависимости от характера решаемых задач изменяется не только порядок выполнения операций, но и морфология модели.

## 1.3. Типовая структура системной модели

Несмотря на большое количество разработок в области моделирования систем различного назначения, пока не создано единой теории и технологии построения системных моделей, реализующей в полной мере концепцию системного гомеостазиса. Тем не менее, представляется возможным на основе анализа и обобщения имеющегося опыта описать структуру типовой системной модели, выделить образующие ее функциональные блоки и сформулировать некоторые общие положения, которые целесообразно положить в основу их построения.

Структурно-функциональная схема типовой системной модели приведена на рис. 1.4.

Системная модель состоит, как правило, из двух основных компонентов – информационного (выделенного на схеме пунктиром) и операционно-лингвистического (выделенного штрихпунктиром).

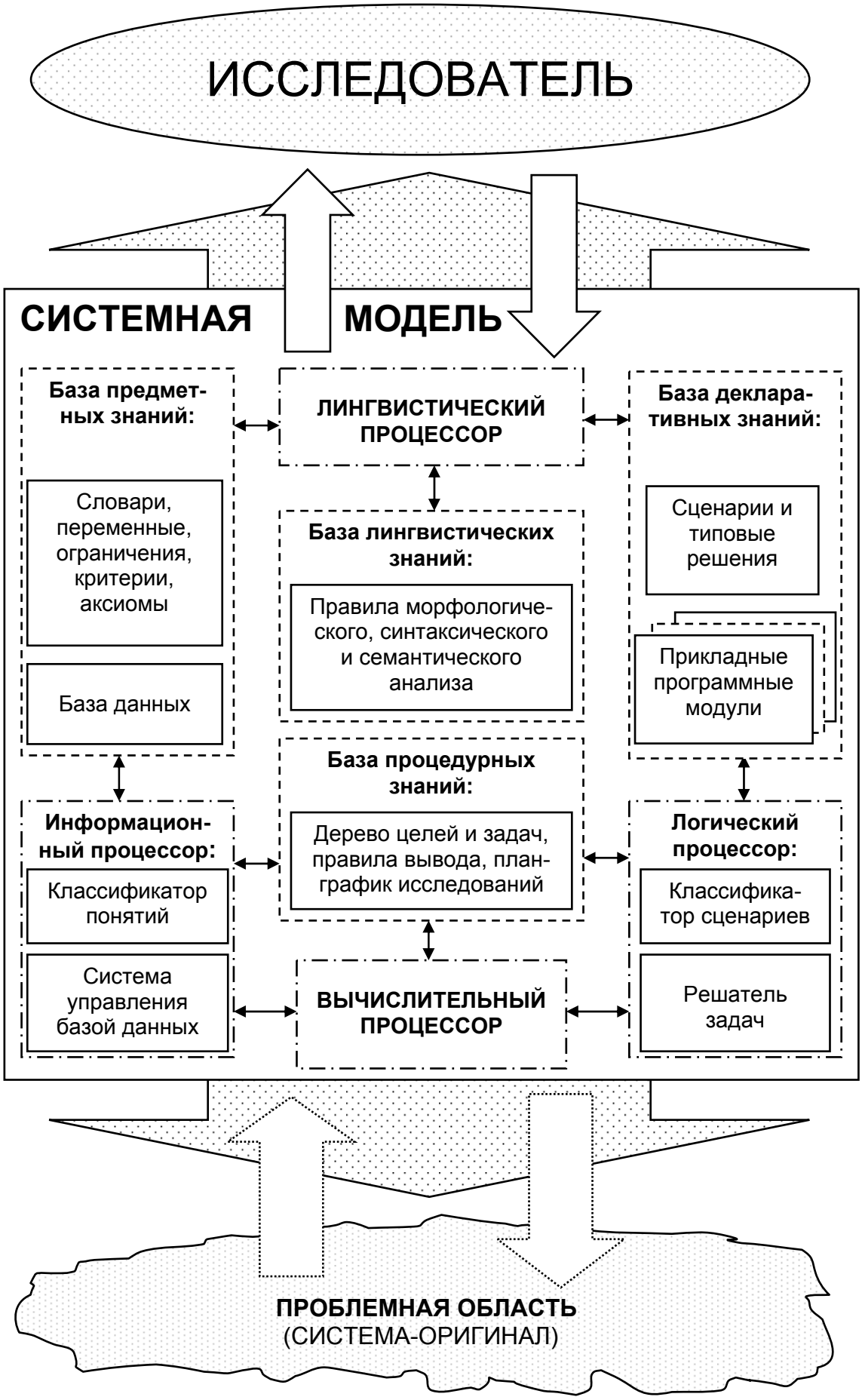


Рис.1.4. *Структурно-функциональная схема типовой системной модели*

### 1.3.1. Информационный компонент

Это структурированное множество информационных объектов, образующих в совокупности знания модели об объекте моделирования, условиях его функционирования и о собственной вычислительной (технической) среде. Иными словами, информационный компонент представляет собой упорядоченные сведения о внешнем и внутреннем мире системной модели. Важнейшим положением, определяющим построение информационного компонента, следует признать идентичность описаний объекта изучения у партнеров, ведущих диалог (исследователя и модели). При этом под идентичностью понимается создание условий, обеспечивающих тождественное понимание информационных объектов, используемых для описания проблемных ситуаций.

В противном случае эффективный диалог невозможен, и компьютер будет использоваться либо в качестве быстродействующей логарифмической линейки, либо как электронная печатающая машинка. Обычно информационный компонент подразделяют на четыре части: базу предметных знаний, базу декларативных знаний, базу процедурных знаний и базу лингвистических знаний.

База предметных знаний – это упорядоченные каким-либо способом факты и данные, отражающие проблемную среду, внешний и внутренний мир модели. Она дифференцируется на пользовательский и прагматический уровни. Пользовательский уровень включает: словари терминов, описания используемых переменных и накладываемых на них ограничений, возможные критерии выбора решений, а также базовые аксиомы – утверждения, которые априори считаются истинными и из которых по определенным правилам выводятся новые утверждения. Прагматический уровень содержит данные о характеристиках изучаемого объекта, условиях его функционирования и о вычислительной среде модели. Фактически, прагматический уровень представляет собой базу данных в ее традиционном понимании.

Базу декларативных знаний образуют правила вывода, на основе которых делают обобщения и заключения. Она имеет также два уровня: пользовательский и прагматический. Пользовательский уровень содержит набор сценариев для анализа ситуаций в объекте изучения, а также набор типовых решений, привязанных к этим ситуациям. Прагматический уровень содержит прикладные программные модули, предназначенные для проведения математических расчетов и вычисления характеристик объекта изучения (в том числе, показателей эффективности). Средняя по сложности системная модель может включать до нескольких сотен прикладных программных модулей, реализующих самые различные методы моделирования (имитационные, оптимизационные, нейросетевые и т.п.).

База процедурных знаний содержит совокупность правил, определяющих порядок и способы применения предметных и декларативных знаний для разрешения проблемы, поставленной пользователем. Ее центральным компонентом является дерево целей и задач, а также правила вывода, в совокупности позволяющие формировать алгоритм поиска решений, например, методом резолюций [Уинстон, 1980]. Кроме того, в процедурной базе содержится план-график или общий алгоритм проведения исследований, определяющий, кто, что, в каком виде и к какому сроку должен сделать.

База лингвистических знаний содержит правила морфологического, синтаксического и семантического анализа входных и выходных текстов, а также списки основ слов, которые используются для организации диалога.

### 1.3.2. Операционно-лингвистический компонент

Этот компонент системной модели представляет собой структурированную совокупность программ, выполняющих процедуры анализа и синтеза текстов естественного языка, запоминания, за-

бывания[*)] и извлечения информации, ее интерпретации, производства логических выводов и принятия решений, а также планирования и оптимизации вычислительного процесса. Операционный компонент, как правило, состоит из четырех основных блоков, которые по сложившейся терминологии называют процессорами.

Лингвистический процессор – это программа, осуществляющая перевод входных текстов естественного языка на язык представления знаний в модели и обратный перевод. Перевод осуществляется в три этапа.

На первом этапе производится морфологический анализ входного предложения с целью идентификации образующих его слов с терминами, используемыми в модели. Для этого используются списки основ слов (слова за вычетом аффиксов – суффиксов и окончаний), которые хранятся в лингвистической базе знаний. Морфологический анализ не является обязательной операцией. В системных моделях, работающих с небольшим словарным запасом входного языка, используют так называемую функциональную лексику, когда каждой фразе присваивается определенный символ, который однозначно воспринимается лингвистическим процессором.

На втором этапе производится синтаксический анализ входной фразы с целью получения формализованной записи синтаксической структуры входного выражения. Для этого используются синтаксические правила, хранящиеся в лингвистической базе знаний, которые позволяют идентифицировать структуру входной фразы со словарем модели.

На третьем этапе производится семантический анализ входной фразы, то есть идентификация ее значения со знаниями модели, хранящимися в предметной и декларативной базах. При нахо-

[*)] Забывание – операция переноса редко используемой информации из оперативной памяти компьютера в долговременную память и, в конце концов, ее стирание как ненужной.

ждении идентичной конструкции входная фраза считается понятной и лингвистический процессор работу заканчивает. Если такой конструкции в базах знаний не содержится, то лингвистический процессор инициирует диалог с пользователем для пополнения или изменения знаний модели. Лингвистические процессоры обычно строятся таким образом, чтобы они удовлетворяли принципу обратимости. Суть этого принципа состоит в том, что все процедуры, обеспечивающие перевод фраз естественного языка в выражения языка представления знаний, должны быть обратимыми. Другими словами, если выходы этих процедур считать их входами и все операции заменить обратными, то те же самые процедуры можно использовать для обратного перевода.

Информационный процессор – это программа, осуществляющая операции запоминания, забывания и извлечения информации о фактах внешнего и внутреннего миров модели. Знания на пользовательском уровне обрабатываются с помощью классификатора понятий, который одновременно участвует в идентификации конструкций входных фраз. Кроме того, информационный процессор реализует функции управления базой данных.

Логический процессор – это программа, осуществляющая классификацию и обобщение текущих ситуаций в объекте исследования, логический поиск рациональных решений и их координацию. Логический процессор участвует в идентификации ситуаций, описываемых входной фразой, с типовыми ситуациями, хранящимися в базе декларативных знаний. При успешной идентификации найденная типовая ситуация передается в решатель, который отыскивает среди эталонных решений (хранящихся в декларативной базе) наилучшее решение для идентифицированной ситуации. Если идентификация ситуации невозможна, производится ее расчленение на более простые, которые могут быть сведены к типовым ситуациям, описанным в базе знаний, либо, напротив, осуществляется ее обобщение с той же целью. Если не удается выделить типовую ситуацию, то процессор формирует

новую ситуацию и передает ее для запоминания в базу знаний. При необходимости процессор координирует отдельные решения.

Вычислительный процессор – это программа, планирующая и выполняющая вычисления. В типовом варианте он состоит из трех программных модулей. Прагматический анализатор на основании описания вычислительной задачи определяет, какие прикладные программные модули необходимо привлечь для ее решения. Кроме того, анализатор определяет состав исходной информации и аппаратных средств, которые должны участвовать в формировании ответа (на рис. 1.4 связь анализатора с вычислительной средой модели не показана). После этого планировщик составляет план вычислительного или логического процесса, оптимальный относительно некоторого критерия, например, минимума времени производства вычислений. Завершает работу вычислительного процессора программатор, который на основании плана автоматически формирует программу вычислений и следит за ходом ее реализации. После завершения программы полученная информация передается в лингвистический процессор, а оттуда – исследователю.

### 1.3.3. Режимы работы системной модели

Принципиальная особенность системных моделей состоит в необходимости их обучения. Начальная программа системной модели содержит лишь операционный компонент, который реализует «интеллектуальные» механизмы модели по преобразованию знаний (в том числе механизмы ее адаптации и интерпретации). Информационный же компонент модели пуст. Для того чтобы работать с моделью, надо сообщить ей факты и закономерности, касающиеся изучаемого объекта, а также план предстоящего модельного эксперимента. В качестве учителя выступает исследователь, который через лингвистический процессор сообщает модели все необходимые знания. В режиме обучения информационный и логический процессоры работают только на прием ин-

формации, а вычислительный процессор вообще не участвует в работе. После обучения модель может работать в вопросно-ответном и интерактивном режиме. В вопросно-ответном режиме исследователь формулирует и задает на вход модели текст, содержащий тему и условия вопроса. Выходом модели являются сведения, удовлетворяющие условиям вопроса и оформленные в виде некоторого текста на естественном языке, либо в виде документа (таблицы, анкеты). В интерактивном режиме (или режиме модельного эксперимента) участвуют все блоки модели. В этом режиме на вход модели поступают описания экспериментальных сценариев, содержащие сведения о целях и задачах данного эксперимента, условиях и графике его проведения и т. д.

Модель инициирует диалог с исследователем до тех пор, пока не получит от исследователя ответов, снимающих всякую неопределенность относительно выполнения предписанных операций. Так, например, если эксперимент заключается в решении некоторой системы дифференциальных уравнений, то модель выдаст исследователю полный перечень вопросов, касающихся граничных и начальных условий, областей определения переменных, необходимой точности вычислений, а также проверит функции, входящие в состав уравнений, на отсутствие точек разрыва.

После снятия вопросов производятся необходимые расчеты, логические выводы и другие операции, позволяющие сформировать на выходе модели рекомендации по изменению параметров объекта изучения. Эта информация в виде таблиц и графиков, сопровождаемых текстами на естественном языке, выдается исследователю. После анализа полученной информации исследователь вводит в модель необходимые корректировки, изменяет условия задачи, и модельный эксперимент продолжается.

### 1.3.4. Классификация системных моделей

Системные модели (рис. 1.5), которые работают только в вопросно-ответном режиме, называются вопросно-ответными или экс-

пертными. Если предметная база содержит лишь прагматический уровень, то есть состоит только из базы данных, то такие модели называются информационно-поисковыми. Модели, работающие в интерактивном режиме, называются информационно-логическими, а в случае, когда объектом управления является некоторая система специального математического обеспечения или пакет прикладных программ, их называют информационно-расчетными.

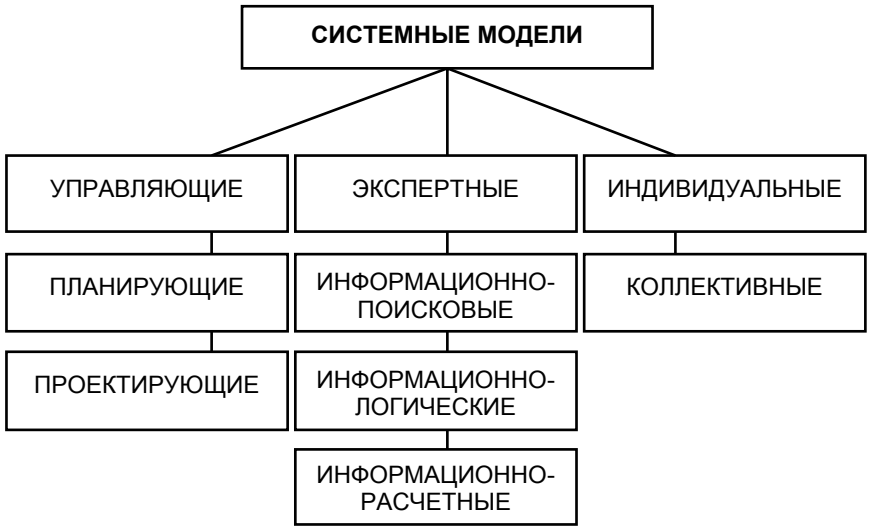


Рис. 1.5. *Классификация системных моделей (вариант)*

В зависимости от предназначения системные модели подразделяются на управляющие, проектирующие и планирующие. Проектирующие модели предназначены для обеспечения проектирования технических, технологических, организационных и других систем, а также для научных исследований. Модели управляющего типа предназначены для управления реальными технологическими, техническими, производственными и другими процессами. Например, такие модели используются в автоматизированных системах управления движением железнодорожного и воздушного транспорта. В том случае, когда системные модели разрабатываются и используются для отработки планов применения каких-либо средств, они называются планирующими.

Индивидуальные модели реализуются на базе локальных компьютерных средств и предназначены для персональной иссле-

довательской, управленческой и планирующей работы. Коллективные системные модели разрабатываются на базе распределенных компьютерных сетей. Они предназначены для проведения комплексных научно-исследовательских и опытно-конструкторских работ в масштабе института или опытно-конструкторского бюро, а также для управления пространственно распределенными объектами. Такие модели входят в состав соответствующих автоматизированных систем управления, образуя их интеллектуальное и информационное ядро.

Конечно, для каждого типа системных моделей характерны свои особенности структурного построения, а также своя специфика в способах программной реализации, но общие принципы их построения примерно одинаковы и соответствуют тому, что рассматривалось выше.

## 1.4. Общая схема модельных исследований

Модельные исследования включают, как правило, три взаимосвязанных составляющих: исследовательскую, технологическую и прагматическую.

### 1.4.1. Исследовательская составляющая

Общая схема проведения модельных исследований приведена на рис. 1.6. Эта схема не нуждается в особых комментариях. Обратим внимание лишь на два обстоятельства.

Во-первых, как это следует из приведенной схемы, системные модельные исследования итеративны по своей сути. Это означает, что с помощью гомеостатической модели любая системная проблема решается путем последовательного приближения и не имеет своего окончательного решения, что согласуется с одним из основных принципов системного подхода – ни при каком сколь угодно глубоком познании невозможно получить исчерпывающую характеристику изучаемого объекта. Задача заключается в том, насколько качественно и в срок данный уровень познания

позволяет разрешить поставленную перед исследователем проблему.

Для оценки качества исследовательского компонента системных модельных исследований используются три группы критериев: категорийно-описательные; количественные и качественные. Категорийно-описательные критерии отражают способность модели выполнять возложенные на нее функции и представляют собой набор и описания таких субхарактеристик и атрибутов, как например: перечень рассчитываемых показателей эффективности и номенклатуру выдаваемой информации; корректность (правильность) получаемых результатов; способность компонентов модели к взаимодействию с исследователем; степень стандартизации интерфейсов; структурно-адаптационные возможности и способность противостоять дестабилизирующим факторам и др.

Количественные критерии применяются для оценки эффективности и надежности модели и характеризуют время ее реакции, объемы используемых информационных ресурсов, завершенность, восстанавливаемость, доступность (готовность), устойчивость к дефектам и т.д. Качественные критерии позволяют оценить модель с точки зрения ее понятности, простоты использования, изучаемости, тестируемости и сопровождаемости; возможности развития и применения в других проблемных областях.

Во-вторых, системные исследования предполагают непременный симбиоз теоретических модельных экспериментов с наблюдениями, эмпирическими исследованиями, натурными (лабораторными) экспериментами. При этом ведущая роль собственно моделирования заключается в том, что модель должна предшествовать натурным экспериментам и указывать направления сбора информации в процессе разного рода наблюдений и эмпирических исследований. В противном случае мы приходим к ситуации, которую образно высказал Сенека: *«Когда корабль не знает, к какой пристани он держит путь, ни один ветер для него не будет попутным»*. В тоже время, результаты модельных исследований

нуждаются в фактических исходных данных и требуют экспериментального подтверждения. Иначе приходим к другой ситуации, которую не менее образно выразил Гете: «*Сера, мой друг, теория везде, златое древо жизни зеленеет*».

### 1.4.2. Технологическая составляющая

Системная модель – это некоторая концептуальная (символьная) система, процесс проектирования и разработки которой органически вписан в общую схему проведения исследований и не существует вне него.

Технология ее проектирования и разработки характеризуется следующими особенностями:

– сложностью предметной области (достаточно большое количество разнообразных функций, объектов, атрибутов и сложные взаимосвязи между ними), требующей комплексного анализа данных и процессов;

– наличием совокупности тесно взаимодействующих разнородных компонентов, имеющих свои локальные цели и задачи функционирования;

– отсутствием прямых аналогов, что ограничивает возможность использования (прямого заимствования) каких-либо типовых проектных решений и прикладных систем;

– функционированием в неоднородной операционной среде на нескольких, как правило, разнородных вычислительных платформах;

– разобщенностью и разнородностью коллективов разработчиков по уровню квалификации и сложившимся традициям использования тех или иных инструментальных средств;

– экономической целесообразностью перманентной модернизации проектируемой модели в интересах решения других (смежных) задач.

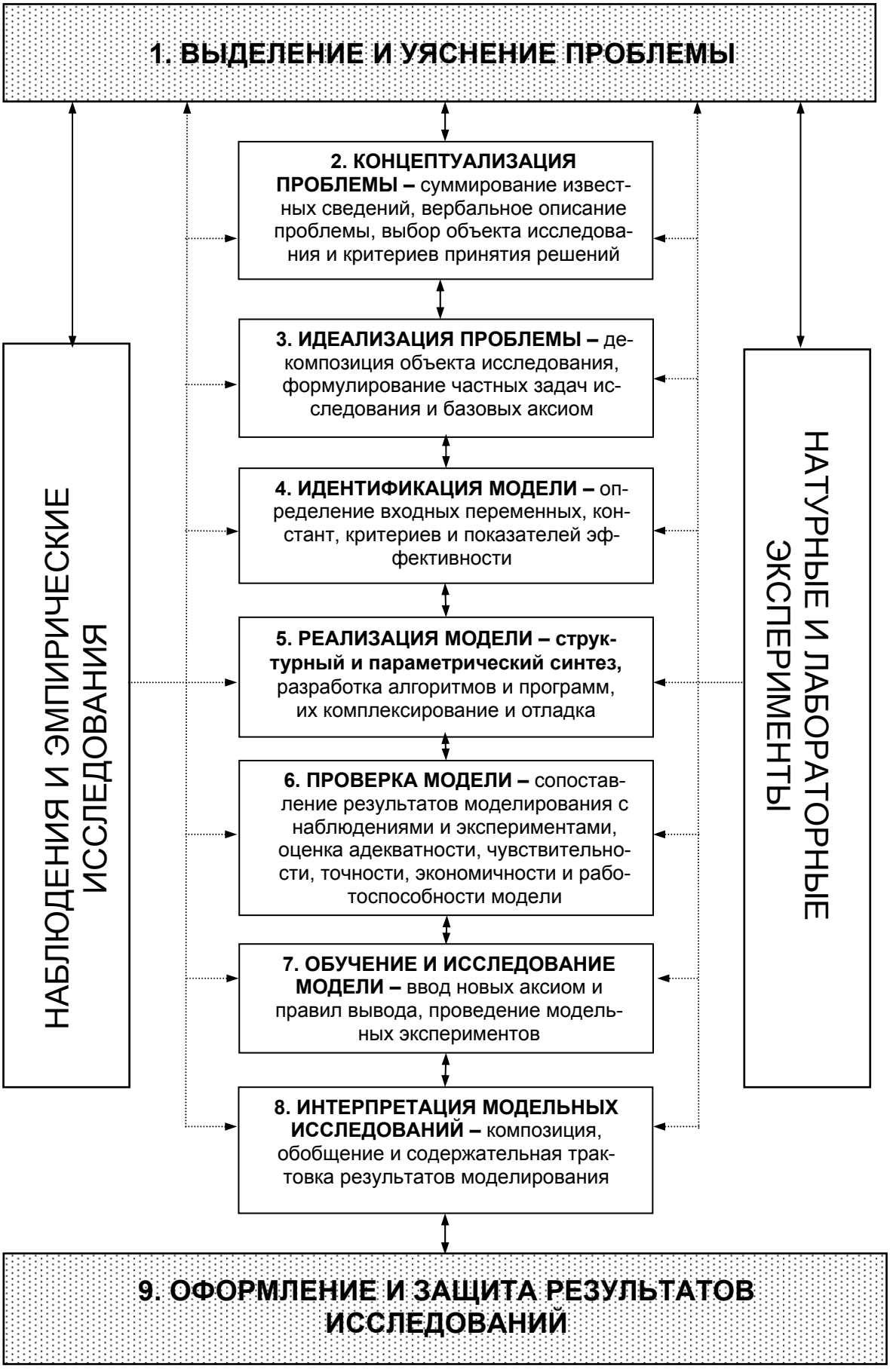


Рис. 1.6. *Общая схема проведения модельных исследований*

Указанные особенности позволяют сделать вывод о том, что для проектирования и разработки системных моделей требуется особая технология, которая должна в полном объеме базироваться на концепциях системного подхода. Рассмотрим кратко основные положения такой технологии.

Подлежащая разработке системная модель должна рассматриваться как сложная не полностью предсказуемая открытая самоорганизующаяся многоуровневая многослойная иерархическая концептуально-техническая система, при проектировании и реализации которой должны соблюдаться принципы преемственности, стандартизации и унификации, сопрягаемости, доступности, интеграции в общую инфраструктуру надсистемы, адаптивности к внешним условиям, апробированности технических, программных и других решений, а также этапности и управляемости.

Основной целью проектирования системной модели является обоснованный выбор рациональных (в определенном потребительском смысле) состава, структуры, параметров и способов функционирования системной модели, отвечающих требованиям технического задания, и удовлетворяющих совокупности объективно существующих ограничений (экономических, производственных, технологических, временных и др.). Это – проблема выбора рационального технического решения для проектируемой модели. Рациональное техническое решение трактуется как устойчивое состояние модели, а выбор рационального технического решения понимается и выполняется как приведение модели к устойчивому состоянию.

Приведение проектируемой модели к устойчивому состоянию трактуется как установление системного гомеостазиса, главной отличительной чертой которого по сравнению с гомеостазисом кибернетической системы является целеориентированность с началом в выделении проблемы и концом в принятии обоснованного технического решения. Целеориентированность системного гомеостазиса задается техническим заданием на модель и позво-

ляет строить весь процесс проектирования как поэтапный итерационный поиск рационального технического решения в заданных условиях и при заданных ограничениях. Вместе с тем техническое задание недопустимо рассматривать как догму. В процессе системного проектирования оно подлежит корректировке по результатам промежуточных работ. Таким образом, системный гомеостазис при проектировании системной модели реализуется в виде самоорганизующегося процесса взаимодействия потребителя и разработчика.

Процесс проектирования системной модели как установление системного гомеостазиса осуществляется на основе формируемых в начале проектирования и непрерывно развивающихся в ходе проектирования методах. Эти методы отражают различные аспекты объекта проектирования и реализуются на компьютерах, например, в виде систем автоматизации проектирования. Формирование, развитие и использование методов проектирования осуществляется в форме диалога разработчиков с компьютером, как интеллектуальным партнером. Главным содержанием такого диалога является логико-лингвистическое, имитационное и оптимизационное моделирование, что в простейшем случае реализуемо в форме взаимочередования известных процедур синтеза структуры и параметрического синтеза.

В тех случаях, когда формализация процессов функционирования отдельных компонентов модели по каким-либо причинам невозможна или нецелесообразна, такой компонент вводится в модель как ее физический элемент (например, в виде эмулятора). При этом системная модель становится смешанной (частично – математической, частично – физической), а сам процесс моделирования будет относиться к типу полунатурного моделирования.

Эффективная реализация на компьютерах комплекса логико-лингвистических, имитационных и оптимизационных моделей возможна только при условии полного и однозначного определения всех категорий и понятий, используемых в моделях, для чего

в процессе проектирования системной модели должен формироваться и применяться системный проблемно-ориентированный язык (СПОЯ). Поскольку каждая теория, используемая при системном проектировании, имеет свой проблемно-ориентированный язык (ПОЯ), то для успеха проектирования необходимо сопряжение СПОЯ с различными ПОЯ, достигаемое за счет обобщающих категорий.

Качество проектирования системной модели в решающей мере зависит от того, насколько полно и правильно учтены, определены и используются в ходе разработки все присущие ей и каждому ее компоненту свойства (положительные и отрицательные, собственные и приобретенные за счет взаимодействия компонентов в составе системы), а также ограничения на действие в системе тех или иных естественно-научных законов, насколько полно выявлены, описаны и применены для целей проектирования автономные законы данной системы. В свою очередь, решение указанных задач зависит от того, насколько правильно определено функциональное пространство модели, метрика, геометрия и группа преобразований этого пространства.

Существенным моментом проектирования системной модели является полнота выявления и описания области неопределенности ее применения, системных инвариантов и автономных законов сохранения. При этом помимо традиционной неопределенности в оценке параметров моделируемого объекта, должна учитываться и неопределенность в оценке будущих условий применения модели, вероятностные неопределенности, а также неопределенности в достоверности внутренних связей модели.

Центральным моментом, наиболее сильно влияющим на качество и устойчивость технического решения, является регулирование перечня приобретаемых моделью и каждым ее компонентом структурных свойств и меры каждого такого свойства с целью удовлетворения требований технического задания. Для этого необходимо не только строго разделять собственные и структур-

ные свойства каждого компонента, добиваясь полноты и неизбыточности этих рядов свойств, но и обеспечивать наилучшую совместимость собственных и структурных свойств каждого компонента модели. Каждый компонент и модель в целом характеризуется вполне определенными механизмами формирования структурных свойств, составляющими ту единственную основу, на которой покоится упомянутое регулирование свойств.

Для проектирования модели как целого важной задачей является формирование полного и неизбыточного перечня ее структурных свойств и введение на этой основе интегральных критериев для оценки и сравнения между собой альтернативных вариантов устройства и функционирования разрабатываемой системы. Успешное решение этой задачи достигается, во-первых, грамотным подходом к целевой декомпозиции модели (построению дерева целей и задач), во-вторых, тем, насколько правильно и полно выявлены, описаны и использованы механизмы согласования локальных целей и задач компонентов с глобальными целями и задачами, решаемыми данной моделью.

Любая системная модель включает одну или несколько баз данных, в которых доминирующее значение приобретают сами данные, их хранение и обработка. Поэтому базы данных целесообразно разделять на два компонента: программные средства системы управления базой данных (СУБД), независимые от сферы их применения и смыслового содержания накапливаемых и обрабатываемых данных; информацию базы данных (БД), доступную для обработки и использования в конкретной проблемно-ориентированной сфере применения. В качестве показателей качества проектируемых баз данных целесообразно использовать: полноту, достоверность, идентичность, актуальность, объем, оперативность, периодичность, глубину ретроспективы, динамичность. Кроме того, важнейшим показателем качества базы данных является защищенность информации.

Основополагающим правилом проектирования системной модели является синтез подходов «от частей к целому» и «от целого к частям», что в решающей мере определяет характер и организацию поэтапных итераций (рис. 1.7). Взаимочередование этих подходов в процессе проектирования модели является главным залогом последовательного конструктивного уточнения задач проектирования и результатов их решения, чем, в первую очередь, обеспечивается устойчивость технического решения.

Проектируемая системная модель должна рассматриваться как многослойная система, в которой в качестве слоев выступают подсистемы технического, программного, лингвистического и

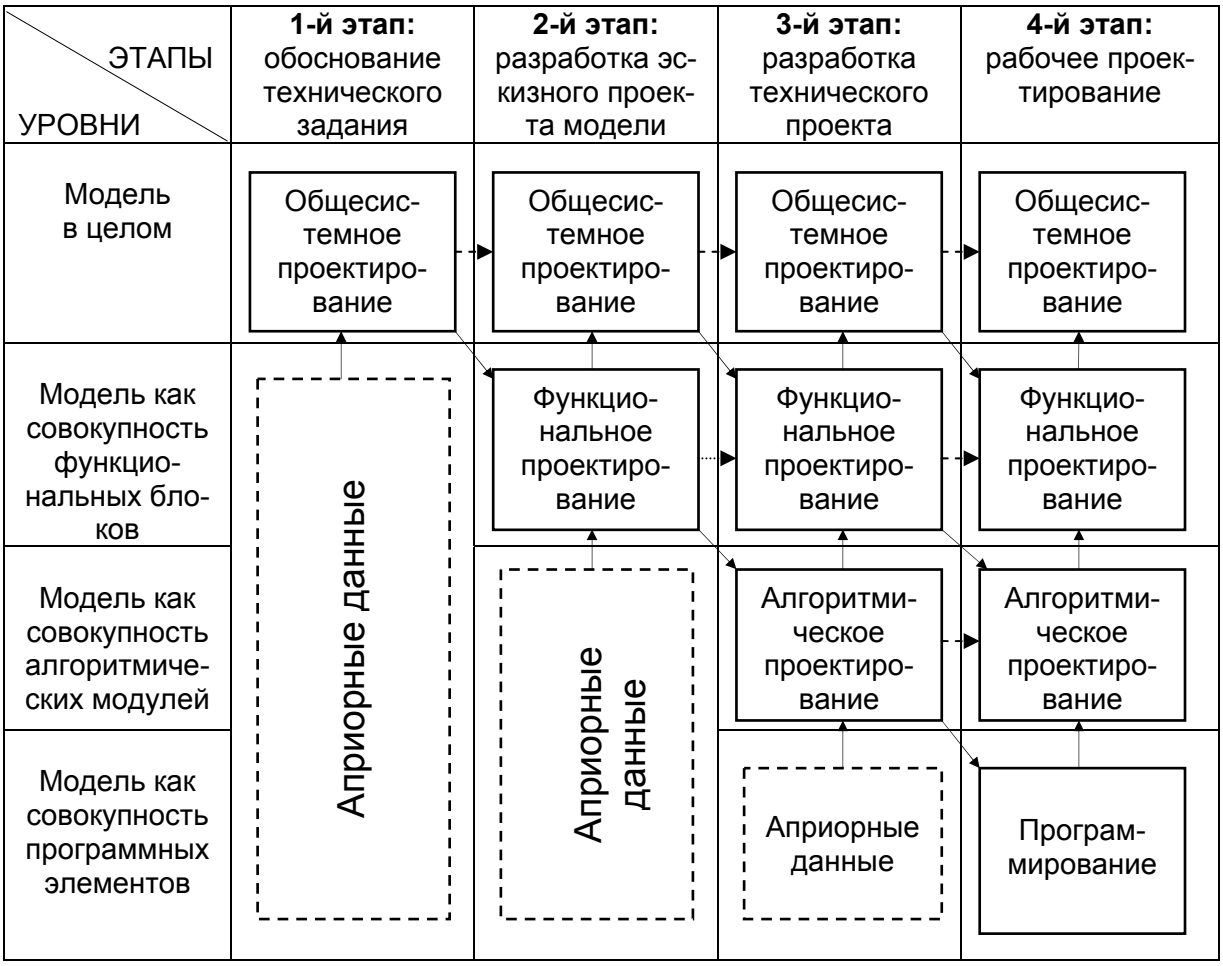


Рис. 1.7. *Технологическая схема проектирования и разработки системной модели (фрагмент)*

информационного обеспечения. Каждая из указанных подсистем имеет свои специфические особенности. Поэтому их проектирование и разработка осуществляется, как правило, специализиро-

ванными коллективами. При этом возникает задача координации их деятельности, иначе, будучи эффективными по отдельности, сложенные все вместе они не обеспечат работоспособность модели в смысле выполнения ею своих основных функций. Решение такой задачи составляет ключевое содержание деятельности руководителя проекта.

Системное проектирование любой системной модели всегда уникально. Элемент научного поиска и творчества здесь играют решающую роль. Однако сколь бы талантливы не были руководитель проекта и члены коллектива разработчиков, необходимо использование специальных методов организации и управления процессом проектирования. В этом отношении наиболее целесообразным следует признать метод сетевого планирования и управления в сочетании с методами планирования и обеспечения качества программных средств (имеются в виду стандарты ISO 12207: 1995; ISO 9000-3:1997; MIL-STD-498, DO-179B).

### 1.4.3. Прагматическая составляющая

Процесс проектирования и разработки любой системной модели связан с разрешением конфликтов трех типов: «потребитель – разработчик», «разработчик – конкуренты» и «разработчик – злоумышленник».

Первый конфликт неантагонистичен по своей сути и разрешается путем поиска компромисса между всегда завышенными требованиями потребителя и всегда ограниченными возможностями разработчика. Центральное место здесь занимает технико-экономическое обоснование проекта, суть которого сводится к сопоставлению ожидаемой прибыли от его реализации и потребных для этого затрат (подробнее см. [Липаев, 2004]).

Конфликт «разработчик – конкуренты» может приобретать самые разнообразные формы от антагонизма до нестрогого соперничества, и даже переходить к различным формам содействия. Соответственно, варьируются и способы его разрешения.

Суть конфликта «разработчик – злоумышленник» заключается в том, что действующая модель является потенциальным объектом воздействия со стороны неких злоумышленников: преступных группировок, хакеров, криминальных образований и др. Разрешить такой конфликт – значит, надежно защитить проектируемую модель от возможных воздействий со стороны злоумышленников. Важно отметить, что решение этой проблемы должно осуществляться не одноактно, а на всех стадиях жизненного цикла модели (от формирования требований к модели до ее сопровождения включительно) путем осуществления комплекса организационных, организационно-технических, технических и программных методов. В общетеоретическом плане это проблема сводится к реализации модели взаимного рефлексивного управления, в которой управляемым объектом выступает процесс проектирования и эксплуатации модели, а рефлексирующими сторонами – разработчик и злоумышленники. При решении проблемы защищенности модели целесообразно ориентироваться на следующие принципы: затраты на создание и эксплуатацию средств защиты модели должны быть не больше, чем размеры возможного ущерба от любых потенциальных угроз; защита программ и данных должна быть комплексной и многоуровневой, ориентированной на все виды угроз; система защиты должна иметь как индивидуальные, так и групповые средства обеспечения безопасности; функционирование системы защиты не должно приводить к существенному снижению эффективности применения модели.

В процессе модельных исследований весьма ответственным является последний (завершающий) этап (блок 9 на схеме рис. 1.6), суть которого заключается в том, что исследователь должен соответствующим образом оформить результаты исследований, представить их заказчику в виде отчета с поясняющими приложениями и, самое главное, защитить их. В процессе защиты заказчик (обычно это комиссия) вправе и должен получить от исследователя ответы на следующие типовые вопросы.

1. Исходя из каких соображений проведение данной работы было поручено данному исполнителю? Проводился ли тендер при заключении договора на выполнение работы?

2. В чем заключалась суть проблемы? Каковы цели проведения исследований, и какие задачи решались для их достижения?

3. Разрабатывался ли план проведения работы? Если да, то на какие этапы она разделялась, каковы сроки выполнения этапов и объемы поэтапного финансирования?

4. Какие сторонние коллективы привлекались для решения задач исследования? Заключались ли субдоговора? Если да, то каковы объемы финансирования субподрядчиков?

5. Какие решения должен принять заказчик по результатам исследования? Когда должны или могут быть приняты эти решения? Какие последствия влекут за собой те или иные решения? Каковы затраты на реализацию предлагаемых решений? Какова ожидаемая прибыль, и в чем она выражается?

6. Какие альтернативные варианты решений были рассмотрены в процессе исследований, и на основе каких соображений осуществлялось их ранжирование? Какие альтернативы были исключены из рассмотрения как заведомо непригодные?

7. Какие исходные предположения составляли базис исследований? Как эти предположения выражены и учтены при построении моделей и проведении исследований? Могут ли некоторые предположения изменить характер выводов? Если да, то каким образом?

8. Какие показатели и критерии использовались при проведении исследований? Каким образом осуществлялся их выбор, и как они согласуются с критериями и показателями более высокого уровня? Существуют ли другие критерии и показатели, которые также представляются разумными?

9. Результаты каких фундаментальных исследований использовались при проведении работы? Чем подтверждается комплексный характер данных исследований?

10. Какие типы неопределенностей учитывались при проведении исследований? Учтены ли помимо традиционных неопределенностей в оценке параметров также неопределенности в оценке будущей обстановки, вероятностные неопределенности, неопределенности в достоверности внутренних связей модели и др.? Если учтены, то каким образом?

11. Учитывались ли при проведении исследований возможности конкурента или противника, или именно такие возможности определяли общий подход к формулированию проблемы и общую технологию исследований?

12. Какие предположения, допущения и ограничения положены в основу построения математических моделей? Какие методы использовались при построении этих моделей? Каким образом оценивалась адекватность моделей, точность и достоверность полученных оценок?

13. Проводились ли в процессе исследований натурные и лабораторные эксперименты? Если да, то как их результаты согласуются с результатами компьютерных экспериментов?

14. Что служит основанием считать данные исследования завершенными? Имеют ли они перспективу? Если имеют, то каковы приоритетные направления дальнейших исследований?

# ГЛАВА 2. ЛОГИКО-ЛИНГВИСТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

Логико-лингвистическое моделирование вошло в практику системных исследований в конце прошлого века как органическое объединение теоретических результатов трех научных направлений: искусственного интеллекта, математической лингвистики и компьютерных языковых технологий. В настоящее время логико-лингвистические модели все шире внедряются в практику системных исследований. Более того, даже специалисты, далекие от системных проблем, стали обращать свое внимание на этот класс моделирования. Этот объективный и закономерный процесс обусловлен, по меньшей мере, двумя обстоятельствами.

Первое обстоятельство связано с тем, что в сферу интересов ученых-практиков все чаще стали попадать объекты, для познания которых либо невозможно построить достаточно адекватную математическую модель, либо модель столь сложна, что с ее помощью не удается выяснить причинные связи между изучаемыми процессами и объяснить те или иные результаты. К числу таких объектов относятся системы самой различной природы, для которых помимо всех остальных черт сложных систем характерны, по крайней мере, следующие свойства:

– большое количество слабо формализуемых и зачастую противоречивых целей функционирования с одновременной их изменчивостью (ситуативностью) во времени;

– конфликтный и многоаспектный характер взаимоотношений как между их компонентами, так и со средой под сильным влиянием человеческого фактора;

– преимущественно понятийный характер исходных описаний условий функционирования и внешних ограничений.

Второе обстоятельство определено настоятельной необходимостью повышения интеллектуального уровня компьютерных технологий. Современные компьютеры (вместе с их программ-

ным обеспечением) достигли столь высокого уровня развития, что на них можно и нужно возложить не только задачи проведения расчетов и выдачи оперативных справок, но выполнение еще целого ряда функций, ранее считавшихся прерогативой человека. А именно: восприятия информации, выраженной в словесной форме; ее структурирования, логического анализа, целеобразования и планирования, формулирования выводов и другие.

В отличие от традиционных математических вычислений, имеющих дело с жестко определенными символьными объектами, отражающими количественные соотношения действительного мира, логико-лингвистические модели оперируют с понятиями, подобными тем, которыми пользуются люди для общения друг с другом и для описания ситуаций. Вместе с тем, в них присутствуют определенные правила формализованного преобразования языковых конструкций, позволяющие говорить об их близости к традиционным вычислительным процедурам и математическим моделям.

## 2.1. Языковые средства логико-лингвистических моделей

Научное исследование предполагает описание фактов и закономерностей изучаемых объектов, анализ полученных описаний и выработку на этой основе рациональных решений, которые затем воплощаются в те или иные действия. Человек производит аналитические операции, используя естественный язык, обладающий большой выразительной способностью в том смысле, что с его помощью можно описывать любые факты и явления окружающего нас мира сколь угодно подробно. Вместе с тем, в естественном языке отсутствуют инструменты, обеспечивающие формальный анализ описаний и определение синтаксической и семантической правильности используемых языковых конструкций (их соответствие принятой аксиоматике). Реализация таких механизмов остается за человеком.

Для традиционных математических языков (арифметических, алгебраических, геометрических, дифференциальных, предикатных и других) характерна крайность другого рода. Они обладают минимальными описательными возможностями, но зато у них в высшей степени развиты инструменты формальных эквивалентных преобразований (стандартные операции и расчетные формулы, теоремы и их следствия, вычислительные и логические алгоритмы), позволяющие на основе заданной аксиоматики формировать практически неограниченное количество правильных в данном языке символьных конструкций.

Элементарным примером такого инструмента могут служить формулы для вычисления площадей и объемов геометрических фигур. Для определения объема или площади фигуры достаточно назвать ее тип и задать параметры (высоту, ширину и т.п.). В том случае, когда нас не интересуют мелкие детали (неровности, сколы, искривления поверхности и др.), вычисления производятся в соответствии с известными формулами, одинаковыми для всех фигур данного типа.

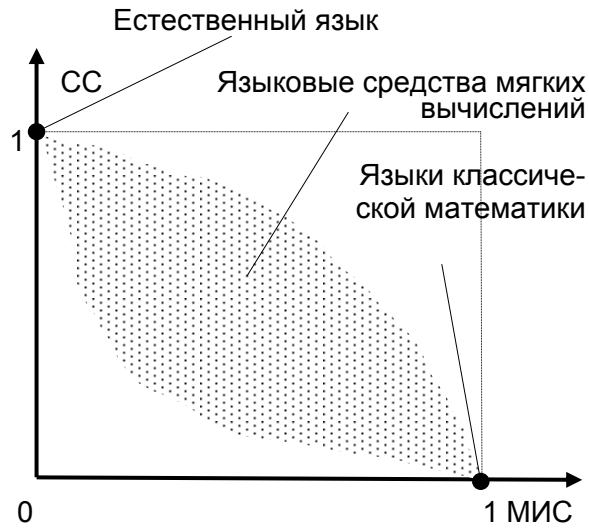

Рис. 2.1. *Сравнительная характеристика языков по семантической силе и мощности инструментальных средств*

На диаграмме рис. 2.1. дается качественное сравнение естественных и математических языков по двум условным характеристикам: семантической силе СС = {0…1}, отражающей возможности данного языка по многоаспектному описанию ситуаций; мощности инструментальных средств МИС = {0…1}, отражающей способность языка производить формально-эквивалентные преобразования своих конструкций.

Из диаграммы видно, что чем выше семантическая сила языка, тем ниже мощность его инструментальных средств, и наобо-

рот – чем выше мощность языковых инструментальных средств, тем выше семантическая сила языка. Наличие противоположных тенденций свидетельствует о существовании некоего компромисса, то есть таких искусственных языковых средств, которые, с одной стороны, позволяют достаточно адекватно описывать проблемные ситуации в их многообразии, а с другой – обладают достаточно развитым инструментарием для производства формальных эквивалентных преобразований.

К таким языковым средствам относятся: нечеткие множества, реляционные и ролевые языки, составляющие в совокупности лингвистическую основу мягких вычислений. Рассмотрим перечисленные языки, имея целью ознакомление с принципами их построения и потенциальными возможностями с точки зрения практических приложений. Более подробные сведения об этих языках можно получить в специальной литературе [см., например, Аверкин, Батыршин и др., 1986; Заде, 1976, Поспелов, 1981; Уэно, Исидзука, 1989].

### 2.1.1. Нечеткие множества

***Понятия и обозначения***. Классическая теория множеств построена на дихотомии (от греч. *dichotomía* – разделение надвое), то есть элемент x может либо принадлежать множеству A, либо не принадлежать ему. В теории нечетких множеств дихотомия отсутствует. Вместо нее принадлежность элемента x множеству A задается функцией принадлежности $\mu_A(x)$.

Нечетким множеством A на множестве X называется множество упорядоченных пар $A = \{\mu_A(x), x\}$, составленных из элементов x универсального (полного) множества X, и соответствующих функций принадлежности $\mu_A(x)$. Переменная x называется базовой. Обычно на нечеткое множество ссылаются либо по его имени A, либо по функции принадлежности.

Если универсальное множество X состоит из конечного числа элементов $x_1, x_2,\ldots, x_n$, то нечеткое множество A можно представить в следующем виде:

$$A = \mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \ldots + \mu_A(x_n)/x_n = \sum_{i=1}^{n} \mu_A(x_i)/x_i. \qquad (2.1)$$

В данном случае знак «+» не есть сложение в его традиционном понимании, им обозначается совокупность элементов множества (знаменатель) с их принадлежностью (числитель).

Функция принадлежности может выражаться как числами из интервала [0, 1], так и в виде лингвистических переменных, то есть переменных, значениями которых выступают не числа, а слова и словосочетания. Например, лингвистическая переменная О = <ошибка> может иметь значения $О_{об}$ = <отрицательно большая>, $О_{ом}$ = <отрицательно малая>, $О_{н}$ = <нуль>, $О_{пм}$ = <положительно малая>, $О_{пб}$ = <положительно большая>.

Таким образом, по определению нечеткие множества представляют собой расширение обычных (строгих) множеств, которые есть не что иное, как частный случай нечетких множеств.

Содержательная трактовка функции принадлежности зависит от задачи, в которой используется нечеткое множество. Возможные трактовки функции принадлежности: степень соответствия понятию А, вероятность, возможность, полезность, истинность, правдоподобность, значение функции.

Для каждой трактовки функции принадлежности разрабатываются свои методы их построения. В ряде моделей мягких вычислений функции принадлежности задаются в параметрическом виде. Например, функции принадлежностей нечетких множеств $О_{об}$, $О_{ом}$, $О_{н}$, $О_{пм}$, $О_{пб}$ могут изначально задаваться в модели так, чтобы они равномерно покрывали область определения X, а затем настраивались в результате изменения их параметров в процессе отладки модели.

В случае, когда функция принадлежности задается числами, наиболее распространенными в приложениях теории нечетких множеств являются линейные, треугольные, трапециевидные, гауссовские и колоколообразные функции принадлежности.

Линейная функция принадлежности задается двумя параметрами

(a, b): $\mu(x) = 0$, при $x \le a$; $\mu(x) = (x - a)/(b - a)$,
при $a < x \le b$; $\mu(x) = 1$, при $x > b$.

Треугольные функции принадлежности задаются тремя параметрами (a, b, c):

$\mu(x) = 0$, при $x \le a$; $\mu(x) = (x - a)/(b - a)$, при $a < x \le b$;

$\mu(x) = (c - x)/(c - b)$, при $b < x \le c$; $\mu(x) = 0$, при $x > c$.

Трапециевидные функции принадлежности задаются четырьмя параметрами (a, b, c, d):

$\mu(x) = 0$, при $x < a$; $\mu(x) = (x - a)/(b - a)$, при $a < x \le b$;

$\mu(x) = 1$, при $b < x \le c$; $\mu(x) = (d - x)/(d - c)$,
при $c < x \le d$; $\mu(x) = 0$, при $x > d$.

Гауссовские функции принадлежности задаются двумя параметрами (c, s):

$\mu(x) = \exp [- 0{,}5(x - c)^2/s^2]$.

Колоколообразные функции принадлежности задаются тремя параметрами (a, b, c): $\mu(x) = 1/\{1 + [(x - c)/a]^{2b}\}$.

Нечеткое множество A нормально, если верхняя граница его функции принадлежности равна единице: $\sup \mu_A(x) = 1$. При $\sup \mu_A(x) < 1$ нечеткое множество называется субнормальным. Нечеткое множество пусто, если $\mu_A(x) = 0$, $\forall x \subset X$. Непустое нечеткое множество нормализуется, то есть приводится к нормальному, по формуле: $\mu_A(x) = \mu_A(x)/\sup \mu_A(x)$.

Множеством уровня $\alpha$ ($\alpha$-срезом) нечеткого множества A называется четкое подмножество универсального множества X, определяемое в виде: $A_\alpha = \{x \subset X \mid \mu_A \ge \alpha\}$, где $\alpha \subset [0, 1]$. От множе-

ства уровня $A_\alpha$ отличается множество строгого уровня $A^*_\alpha$, которое определяется в следующем виде: $A^*_\alpha = \{x \subset X \mid \mu_A(x) > \alpha\}$.

Всякое нечеткое множество можно разложить по уровням:

$$\mu_A = U(\alpha, \mu_A^\alpha),$$

где $\mu_A^\alpha = 1$, если $\mu_A(x) \geq \alpha$ и $\mu_A^\alpha = 0$, если $\mu_A(x) < \alpha$.

Это разложение лежит в основе другого способа задания нечеткости, когда она выражается набором иерархически упорядоченных четких множеств.

Рассмотрим примеры задания нечетких множеств. Пусть X – множество всех мужчин, различающихся только по своему росту; x – переменная, означающая конкретного мужчину. Требуется задать нечеткое множество A высоких мужчин. По определению нечеткого множества для этого достаточно выбрать из каких-либо соображений функцию принадлежности и задать ее параметры. Выберем для определенности гауссовскую функцию принадлежности, и будем полагать, что высокими считаются мужчины, рост которых составляет 180 – 190 см. Тогда c = 185, s = 5, а искомое нечеткое множество запишется в следующем виде: $A = \{\exp[-0{,}002(x - 185)^2], x\}$. Такая запись может, например, означать, что если перед нами находится некий мужчина, рост которого составляет 175 см, то он с уверенностью 82 % может быть назван высоким мужчиной, а некто с ростом 165 см относится к высоким мужчинам с уверенностью лишь 0,3 %.

Другой пример. Пусть универсальное множество – это множество людей в возрасте от 0 до 90 лет, а функция принадлежности нечетких множеств, означающих возраст: «молодой», «средний», «пожилой» определяется так, как показано на рис. 2.2. Тогда при дискретизации 10 лет получаем следующее:

$<\text{молодой}> = \mu_{\text{молодой}}(x) = 1/0+0{,}7/10+0{,}5/20+0{,}3/30+0/40;$

$<\text{средний}> = \mu_{\text{средний}}(x) = 0/20+0{,}5/30+1/40+0{,}5/50+0/60;$

$<\text{пожилой}> = \mu_{\text{пожилой}}(x) = 0/40+0{,}25/50+0{,}4/60+0{,}6/70+0{,}75/80+1/90.$

Очевидно, что нечеткое множество несет в себе не количественную, а качественную смысловую нагрузку, отражающую неоднозначное восприятие ситуации конкретным субъектом. Эта неоднозначность формально выражается функцией принадлежности, поэтому она есть визитная карточка нечеткого множества, а все операции над нечеткими множествами есть операции над их функциями принадлежности.

Оперируя нечеткими множествами, мы, как и ранее, имеем дело с числами. Но в данном случае числа не несут метрическую смысловую нагрузку, то есть, не выражают количественное свойство какого-либо объекта. Примеры таких чисел: номер троллейбусного маршрута, оценка по успеваемости, номер на майке спортсмена и т. д. Это – числа-имена. К ним неприменимы обычные арифметические операции, такие как сложение, вычитание, сравнение и другие, а если подобные операции производятся, то получаются абсурдные по своему смыслу результаты. Это очевидное обстоятельство часто игнорируют, и на вопрос: «Какой дом выше – пятиэтажный или трехэтажный?», – с уверенностью отвечают – «Конечно, пятиэтажный». Или другой вопрос, аналогичного свойства: «Какой научный работник более профпригоден – написавший за год двадцать статей или одну статью?». Кто не чувствует смысловой идентичности этих вопросов и в своей деятельности использует тестирование, тот либо пытается, как в известной сказке Б. Заходера, измерить длину удава в попугаях, либо сознательно искажает действительность, преследуя свои личные интересы.

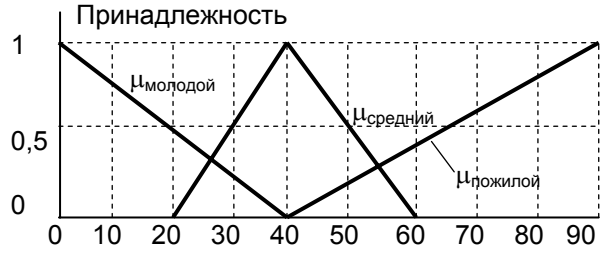

*Рис. 2.2. Пример графического задания нечеткого множества*

За всякого рода тестированиями чаще всего не стоит стремление установить истину, а, скорее всего, скрывается подоплека политического, идеологического, конъюнктурного или какого-либо

другого характера, маскирующаяся научной терминологией. Предпринимаемые попытки введения в тесты нечеткости не позволяют исключить тенденциозность в трактовке их результатов, однако несколько смягчают категоричность выводов, а самое главное, заставляют задуматься над смыслом самого тестирования. Если для изучения какого-либо явления не удается предложить ничего другого, кроме тестирования, то полезно помнить, что получаемая при этом информация может служить лишь пищей для размышления, но не основанием для принятия ответственных решений. То же самое относится к попыткам прямого разрешения системных проблем методом опроса экспертов.

Этот метод является антитезой системному моделированию и исходит из предположения, что существуют эксперты, способные правильно разрешать проблемы определенного класса. Тогда достаточно собрать совет экспертов, ввести их в курс дела, сформулировать вопросы и дождаться ответов. Так зачастую и поступают, забывая о том, что тем самым фактически слагают с себя проблему многоаспектного анализа ситуации, перепоручая ее решение, может быть, компетентным, но посторонним людям. Адмирал Нахимов по такому случаю говорил: «*Ум хорошо-с, два лучше-с, ну и зови одного-с, а то накличут целую сотню, кричат, шумят-с, говорят вздор-с, потом закусят и разойдутся-с, позабыв зачем приходили-с. Для военных советов-с я раз навсегда болен-с!*» [Макаров, 1942]. Оказывается, что найти этого «*одного-с*» очень и очень трудно.

***Операции над нечеткими множествами.*** Простейшей является операция дополнения нечеткого множества, выступающая аналогом связки «не» (отрицания) в математической логике. Символическая запись дополнения для $\forall x \subset X$: $\overline{\mu}(x) = 1 - \mu(x)$. На рис 2.3 приведена графическая иллюстрация дополнения нечеткого множества. Следующей является операция разности нечетких множеств:

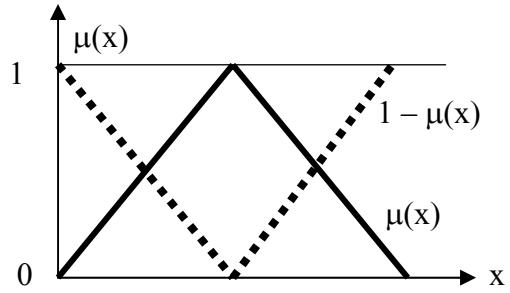


Рис. 2.3. *Дополнение нечеткого множества*

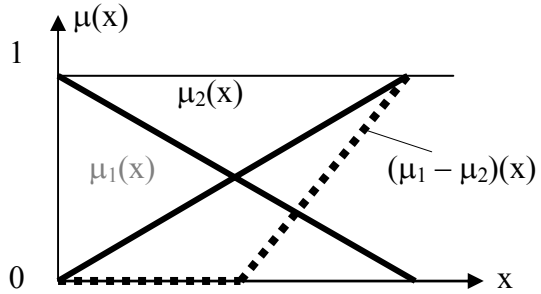


Рис. 2.4. *Разность нечетких множеств*

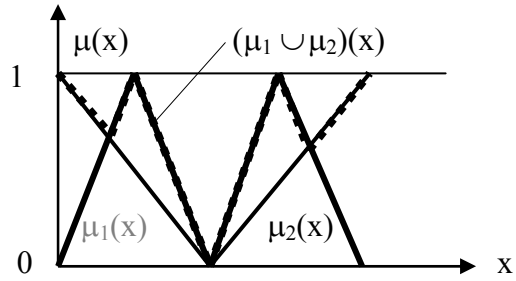


Рис. 2.5. *Объединение нечетких множеств*

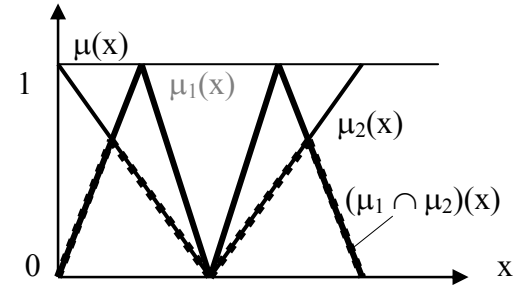


Рис. 2.6. *Пересечение нечетких множеств*

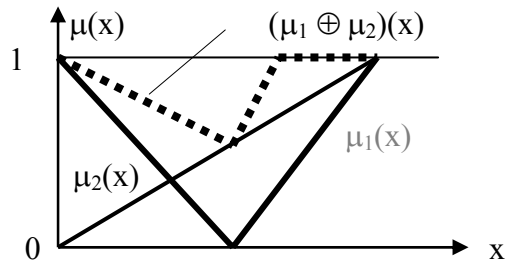


Рис. 2.7. *Ограниченная сумма нечетких множеств*

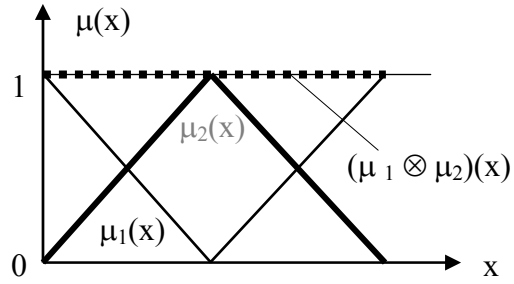


Рис. 2.8. *Ограниченное произведение нечетких множеств*

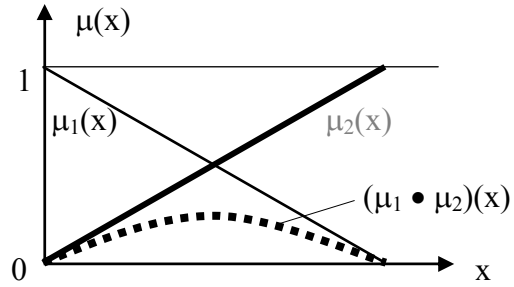


Рис .2.9 *Алгебраическое произведение нечетких множеств*

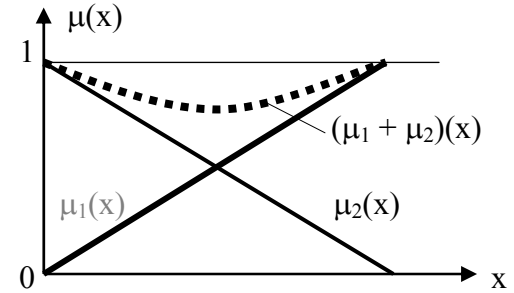


Рис.2.10. *Алгебраическая сумма нечетких множеств*

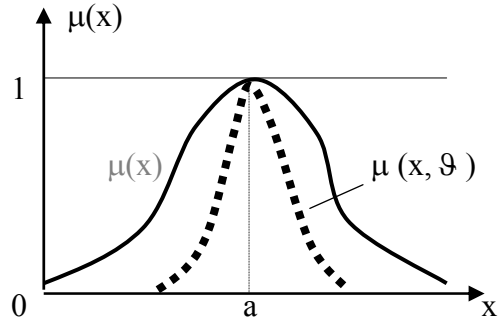


Рис. 2.11. *Концентрирование нечеткого множества*

$$(\mu_1 - \mu_2)(x) = \max\{0, \mu_1(x) - \mu_2(x)\},$$

где индекс 1 относится к уменьшаемому множеству, а индекс 2 – к вычитаемому (рис. 2.4).

Объединение нечетких множеств (рис. 2.5) – максимум (аналог логическому «или» как «либо-либо»), записывается так:

$$(\mu_1 \cup \mu_2)(x) = \max\{\mu_1(x), \mu_2(x)\}.$$

Поясняемое иллюстрацией на рис. 2.6, пересечение нечетких множеств (минимум, аналог логическому «и» как «и-и») записывается так:

$$(\mu_1 \cap \mu_2)(x) = \min\{\mu_1(x), \mu_2(x)\}.$$

Следующая операция – ограниченная сумма нечетких множеств (логическое «или») – поясняется графиками, приведенными на рис. 2.7, а формально записывается следующим образом:

$$(\mu_1 \oplus \mu_2)(x) = \min\{1, \mu_1(x) + \mu_2(x)\}.$$

Ограниченное произведение нечетких множеств (логическое «и») иллюстрируется графиками рис. 2.8 и формально так:
$(\mu_1 \otimes \mu_2)(x) = \max\{0, \mu_1(x) + \mu_2(x)\}$.

Алгебраическое произведение нечетких множеств записывается так:

$$(\mu_1 \bullet \mu_2)(x) = \mu_1(x) \bullet \mu_2(x),$$

а графическая иллюстрация этой операции приведена на рис. 2.9.

Алгебраическая сумма нечетких множеств (логическое «или») формально записывается выражением:

$$(\mu_1 + \mu_2)(x) = \{\mu_1(x) + \mu_2(x)\} - \mu_1 \bullet \mu_2)(x),$$

а графически иллюстрируется графиками на рис. 2.10.

Необычной по сравнению с четкими множествами является операция концентрирования нечеткого множества, смысл которой поясняется графиками на рис. 2.11. Например, для гауссовской функции принадлежности операция концентрирования не-

четкого множества может быть записана в виде следующего выражения:

$$\mu\,(x,\,\vartheta) = \exp\,[-\,0{,}5\,(x-a)^2/\vartheta\,b^2],$$

где $\vartheta = (0,\ldots,1)$ – коэффициент концентрирования. При $\vartheta \to 0$ происходит «сжатие» функции принадлежности, то есть возрастает степень принадлежности x, близких к «точке» a, множеству X. При $\vartheta \to 1$ происходит восстановление первоначально заданной функции принадлежности. Операция концентрирования используется для настройки и отладки нечетких алгоритмов.

Перечень операций над нечеткими множествами не ограничивается указанными. Он может расширяться за счет ввода новых операций, а сами операции могут модифицироваться в зависимости от свойств изучаемых объектов.

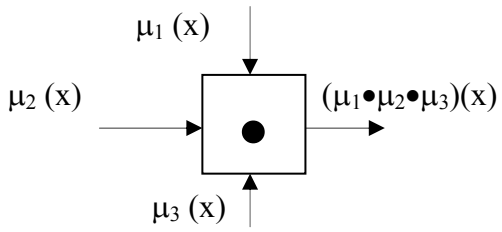


Рис. 2.12. *Пример автоматного представления алгебраического произведения нечетких множеств*

Операции над нечеткими множествами можно интерпретировать простейшим абстрактным автоматом – преобразователем информации, который выдает определенный выходной сигнал в ответ на каждый входной сигнал. В качестве примера на рис. 2.12 изображен абстрактный автомат, имеющий три входа и один выход и соответствующий операции алгебраического произведения трех нечетких множеств $\mu_1\,(x)$, $\mu_2\,(x)$ и $\mu_3\,(x)$. Работает такой автомат просто: в каждый момент $t_i$ дискретного времени он либо принимает входные сигналы $\mu(x)$, либо порождает выходной сигнал $(\mu_1 \bullet \mu_2 \bullet \mu_3) = \mu_1(x) \cdot \mu_2(x) \cdot \mu_3(x)$. Автоматное представление нечетких множеств удобно для составления нечетких алгоритмов.

***Нечеткие алгоритмы.*** Нечеткими алгоритмами называются упорядоченные совокупности операций над функциями принадлежности нечетких множеств. Они описывают сложные отноше-

ния между нечеткими множествами и бывают с фиксированной и нефиксированной структурой. Алгоритмом с фиксированной структурой однозначно предписывается последовательность выполнения элементарных операций над функциями принадлежности нечетких множеств, выступающими исходными данными. Пример нечеткого алгоритма А с фиксированной структурой приведен на рис. 2.13, а его формальная запись представляется в следующем виде:

$$\mu(A)(x) = \min \{1, \min [\mu_1(x), \mu_2(x)] + \max [0, \mu_1(x) - \mu_3(x)]\}. \quad (2.2)$$

При заданных функциях принадлежности $\mu_1(x)$, $\mu_2(x)$ и $\mu_3(x)$ такая запись однозначно определяет порядок выполнения элементарных операций и позволяет установить совокупное нечеткое множество $\mu(A)(x)$. Алгоритмы с четко фиксированной структурой используются для моделирования детерминированных процессов.

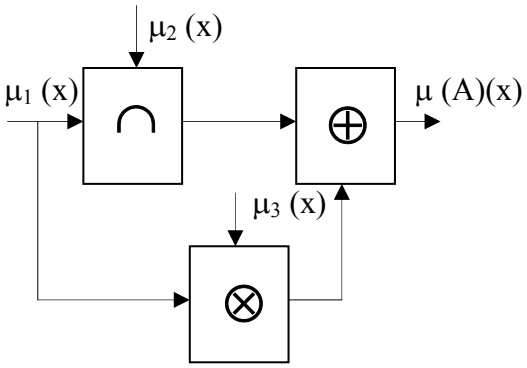


Рис. 2.13. *Пример графического задания нечеткого алгоритма с фиксированной структурой*

Для моделирования полифуркационных (ветвящихся) процессов используются алгоритмы с нечетко фиксированной структурой. В этих алгоритмах не предписывается цепочка действий, а указываются лишь возможные (разрешенные) последовательности выполнения элементарных операций. Они строятся на основе простых условных операторов и условных операторов с обратной связью.

Простой условный оператор имеет вид, представленный на рис. 2.14, и формально записывается следующим образом:

$$O(x, \theta, \lambda): \lambda \Rightarrow \{[y_1 = \theta_1 \mu(x)], \ldots, [y_N = \theta_N \mu(x)]\}, \sum \theta_i = 1. \quad (2.3)$$

Он имеет тот смысл, что если выполняется некоторое условие $\lambda$, то на выходе с номером i (i = = 1, …, N) вырабатывается

сигнал $y_i = \mu(x)$ с некоторой вероятностью $\theta_i$ от 0 до 1. Если условие $\lambda$ не выполняется, то на всех выходах сигнал отсутствует, $y_1 = \ldots = y_N = 0$. В частном случае, когда $\lambda$ принимает только два значения, например 0 и 1, простой условный оператор будет детерминированным.

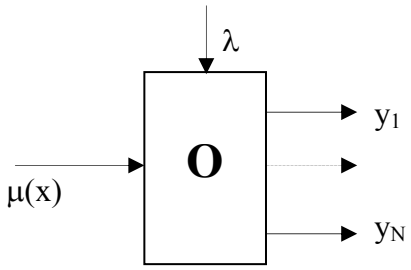


Рис. 2.14. *Простой условный оператор*

Условные операторы с обратной связью используются при разработке алгоритмов, моделирующих адаптивные и самоорганизующиеся процессы. На рис. 2.15 приведен один из возможных вариантов такого оператора, реализующего обратную связь типа ограниченного произведения $\otimes$ по выходу $y_N$. Изменяя конфигурацию такого оператора и варьируя операциями, образующими контур обратной связи, можно имитировать функционирование систем как с положительными, так и с отрицательными обратными связями (детерминированными, случайными, запаздывающими и другими).

Моделирование изучаемых процессов с использованием нечетких алгоритмов образно можно сравнить с построением различных фигур из «кубиков» конструктора. Операция в принципе простая, но для того чтобы в итоге конструирования получилась требуемая конфигурация, необходимо:

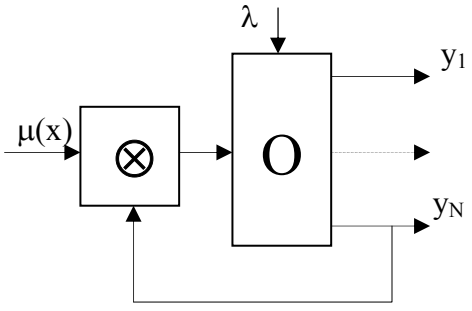


Рис. 2.15. *Условный оператор с обратной связью*

а) иметь конструкторскую схему, то есть достаточно полную понятийную (вербальную) модель объекта моделирования;

б) располагать полным набором совмещаемых между собой «кубиков», то есть задать функции принадлежности нечетких

множеств, которые будут использоваться при конструировании алгоритма.

Зачастую выполнить эти условия не представляется возможным именно потому, что нет полной ясности относительно свойств и возможных вариантов поведения изучаемого объекта. В таких случаях организуется пошаговая разработка алгоритма и его поэтапная апробация с «обучением». Учителем выступает исследователь, а обучение заключается в уточнении и видоизменении первоначально выбранных функций принадлежности, а также структуры алгоритма (используемых операторов, вариантов их соединения и т. п.).

Процесс обучения может завершиться тем, что построенный алгоритм будет неадекватным реальному процессу. Отрицательный результат – тоже результат, добавляющий знания. Во всяком случае, даже неудачные попытки моделирования нечеткими алгоритмами изучаемого процесса всегда полезнее, чем поверхностные наблюдения, сомнительные аналогии и умозрительные заключения.

***Нечеткие отношения.*** Как известно из теории обычных множеств, отношением между множествами X и Y называется подмножество R, образованное прямым произведением множеств X и Y, $R \subseteq X \times Y$. По аналогии, нечеткое отношение между множествами X и Y можно задать функцией принадлежности: $\mu_R$: $X \times Y \rightarrow L$, где L – множество вещественных чисел, отрезок прямой [0,1], множество лингвистических переменных, полная дистрибутивная решетка и т. п. Ограничимся для простоты только бинарными нечеткими отношениями между множествами X и Y (например, «X примерно равен Y»), понимаемыми как функция R: $X \times Y \rightarrow L$, где L – полная дистрибутивная решетка, то есть частично упорядоченное множество, в котором любое непустое подмножество имеет наибольшую нижнюю и наименьшую верхнюю границы, а операции пересечения $\cap$ и объединения $\cup$ в L удовле-

творяют закону дистрибутивности. Введенное ограничение удобно тем, что позволяет все операции над нечеткими отношениями определять этими ($\cap$ и $\cup$) операциями из L. Например, если под L понимается ограниченное множество вещественных чисел, то операциями взятия наибольшей нижней и наименьшей верхней границ будут операции inf и sup соответственно, а $\cap$ и $\cup$ – min и max соответственно. В простом случае, когда L считается отрезком [0, 1], формальные определения $\mu_R(x, y)$ и R(x, y) идентичны. В частности, если множества X и Y конечны, нечеткое отношение между ними можно представить матрицей отношения. Пример такой матрицы показан в таблице 2.1. Ее строки и столбцы определяют соответствие элементов множеств X и Y, а на пересечениях записываются значения R(x, y). При совпадении множеств X и Y нечеткое отношение R: X $\times$ X $\rightarrow$ L является нечетким отношением на множестве X. Ему соответствует взвешенный граф, пример которого представлен на рис. 2.16. В этом графе вершины соединяются взвешенными ребрами R(x, x).

Т а б л и ц а 2.1

| R | $y_1$ | $y_2$ | $y_3$ |
|---|---|---|---|
| $x_1$ | 0 | 0,4 | 0,2 |
| $x_2$ | 0,4 | 0,6 | 0,5 |
| $x_3$ | 0,2 | 0,5 | 0,9 |

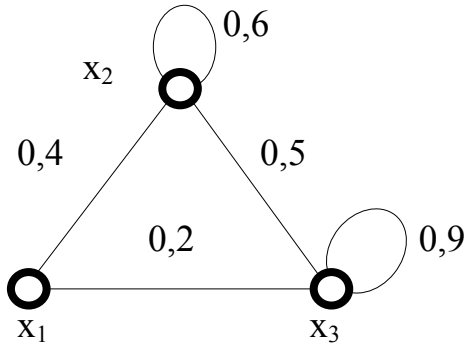


Рис. 2.16. *Взвешенный граф, описывающий нечеткие отношения*

***Операции над нечеткими отношениями.*** Пусть R, Q и T – три нечетких отношения. Если x $\subset$ X, y $\subset$ Y, то операции объединения и пересечения нечетких отношений определяются следующим образом:

$$(R \cup Q)(x, y) = R(x, y) \cup Q(x, y), (R \cap Q)(x, y) = R(x, y) \cap Q(x, y).$$

Операция включения $R \subseteq Q$ определяется с помощью отношения частичного порядка в L: $R \subseteq Q \Rightarrow R(x, y) \leq Q(x, y)$.

Множество $F(X \times Y)$ всех нечетких отношений между X и Y образует дистрибутивную решетку по отношению к операциям объединения и пересечения, если удовлетворяются следующие тождества:

$$R \cap R = R,\ R \cup R = R \text{ (идемподентность);}$$

$$R \cap Q = Q \cap R,\ R \cup Q = Q \cup R \text{ (коммутативность);}$$

$$R \cap (Q \cap T) = (R \cap Q) \cap T,\ R \cup (Q \cup T) = (R \cup Q) \cup T$$
(ассоциативность);

$$R \cap (Q \cap R) = R,\ R \cup (Q \cup R) = R \text{ (поглощение);}$$

$$R \cap (Q \cup T) = (R \cup Q) \cap (R \cup T),\ R \cap (Q \cap T) = (R \cap Q) \cap (R \cap T)$$
(дистрибутивность).

На множестве $F(X \times Y)$ выполняется также следующее соотношение: из $Q \subseteq T$ следует $R \cup Q \subseteq R \cup T$, $R \cap Q \subseteq R \cap T$, а также определяются пустое $\varnothing$ и универсальное U отношения $\varnothing(x, y) = 0$ и $U(x, y) = 1$ для $\forall x \subset X$, $\forall y \subset Y$, для которых справедливо: $R \cap \varnothing = \varnothing$, $R \cup \varnothing = R$ и $R \cap U = R$, $R \cup U = U$.

Для нечетких отношений можно записать и композицию $R \circ Q$ нечеткого отношения R между X иY и нечеткого отношения Q между Y и Z: $(R \circ Q)(x, z) = \mu [R(x, y) \cap Q(y, z)]$, $\forall x \in X$, $\forall z \in Z$.

Теория нечетких множеств охватывает значительное многообразие других вариантов задания композиции нечетких отношений, а также богатую гамму свойств рефлексивности, антирефлексивности, симметричности, транзитивности. В этих категориях можно исследовать сходства и различия нечетких отношений, устанавливать отношения порядка и т. д.

Изложенный подход к заданию нечетких отношений демонстрирует богатые возможности по конструированию языковых механизмов формальных эквивалентных преобразований, которые открывает теория нечетких множеств. Однако вводимые базовые формализмы могут не соответствовать реалиям. В тех слу-

чаях, когда это очевидно, необходимо либо менять исходную аксиоматику, либо ограничиваться алгоритмическим (неаналитическим) способом задания нечетких отношений. В последнем варианте хотя и теряется общность подхода, но зато повышается адекватность модели объекту изучения.

Формальные операции над нечеткими отношениями в прикладных исследованиях справедливы постольку, поскольку отражают свойства изучаемой системы. Это утверждение означает, что при моделировании конкретной системы первичными должны выступать реальные отношения, существующие между ее компонентами, свойства которых следует выражать формальными операциями над нечеткими отношениями. В то же время эти свойства не всегда заранее известны. Поэтому полезна формализация, которую необходимо рассматривать как рабочую гипотезу, подтверждаемую, отвергаемую или уточняемую в ходе исследований.

### 2.1.2. Реляционные языки

Появление реляционных языков (от англ. *relation* – отношение) связано с развитием методов обработки данных на компьютерах. Они явились результатом последовательных попыток усовершенствовать и расширить автоматизацию тех видов обработки информации, которые выполняются человеком. В качестве представителей языков такого типа рассмотрим табличный язык, RX – коды и семантические сети, демонстрируя принципы их построения на примерах из военной области.

***Табличный язык*** представляет сегодня скорее исторический, нежели теоретический интерес, но на практике он используется, и довольно широко. Продемонстрируем возможности такого языка на условном примере. Предположим, что нас интересуют данные, касающиеся характеристик радиоэлектронных средств, находящихся на вооружении армии какого-либо государства. Такого рода данные представляют интерес для разработчиков средств ра-

диосвязи и специалистов по радиоэлектронной борьбе. Эти данные собираются из различных источников, структурируются и вводятся в память компьютеров (табл. 2.2).

Таблица 2.2

**Характеристики радиоэлектронных средств (пример)**

| Шифр средства | Атрибуты | | | | | |
|---|---|---|---|---|---|---|
| | Назначение | Звено управления | Диапазон рабочих частот (МГц) | Излучаемая мощность (Вт) | Дальность действия (км) | Носитель |
| AN/PRC-88 | Радиосвязь | Тактическое | 47 – 57 | 0,3 | 0,5 | Солдат |
| AN/PRC-25 | Радиосвязь | Тактическое | 30 – 76 | 1,5 – 2,0 | 8 – 25 | Танк, БТР |
| AN/GRC-125 | Радиосвязь | Тактическое | 30 – 76 | 1,5 – 2,0 | 8 –25 | Автомобиль |

*Примечание:* Типовой объем подобных таблиц составляет около 10 тыс. шифров радиоэлектронных средств с числом атрибутов более 10.

Данная таблица определяет сущности под общим названием «радиоэлектронные средства, состоящие на вооружении армии определенного государства». Другими примерами сущностей могут быть «штатно-должностной состав воинских подразделений», «автотранспортные средства», «носители высокоточного оружия», «автоматизированные системы управления войсками и оружием» и другие. В первом столбце таблицы записываются имена сущностей, а в остальных столбцах – их атрибуты. Каждый представитель сущности может идентифицироваться двояко. Его можно выделить по индивидуальному имени или по перечню значений атрибутов.

В общем случае каждую строку таблицы можно задать вектором следующего вида: $\langle i; j_1(h_1); j_2(h_2);\ldots;j_m(h_m)\rangle$, где i соответствует имени сущности, $j_m$ – именам атрибутов, а $h_m$ – значениям атрибутов. Подобные представления оказались весьма эффективными при построении компьютерных баз данных, поскольку да-

ют возможность задавать вопросы относительно имен сущностей, атрибутов и их значений. Например, при наличии в памяти компьютера табл. 2.2 можно сформулировать следующие вопросы:

1. ⟨ ? (имя сущности); назначение (радиосвязь); звено управления (тактическое); диапазон частот (47-57); мощность (0,3); дальность действия (0,5); носитель (солдат)⟩ ;

2. ⟨ ? (имя сущности); назначение (радиосвязь); звено управления (тактическое); диапазон частот (λ); мощность (λ); дальность действия (λ); носитель (λ);⟩ ;

3. ⟨ AN/GRC – 125; $\mu_1$(?); $\mu_2$(?); $\mu_3$(?); $\mu_4$(?); $\mu_5$(?); $\mu_6$(?)⟩ ;

4. ⟨ ? (имя сущности); назначение (радиосвязь); звено управления (тактическое); диапазон частот (?); мощность (?); дальность действия (λ); носитель (солдат)⟩ .

Первый пример трактуется как вопрос об имени сущности, которая является носимым средством тактической радиосвязи в диапазоне 47-57МГц, мощностью 0,3Вт и дальностью действия 0,5км. Простая процедура поиска по совпадению всех данных, имеющихся в вопросе, с данными, содержащимися в табл. 7.2, позволяет компьютеру выдать ответ: AN/PRC-88. В формулировке второго примера содержится специальная переменная λ, смысл которой состоит в указании на незначимость атрибутов (в нашем случае «диапазон частот», «мощность», «дальность действия» и «носитель»).

То есть, с получением такого вопроса компьютер должен назвать все средства тактической радиосвязи независимо от их характеристик: AN/PRC-88, AN/PRC-25 и AN/GRC-125. В третьем примере указано только имя сущности, а на остальных местах стоят символы $\mu_j$ (?). Смысл вопроса может быть передан фразой: «сообщить все, что известно о AN/GRC-125». Ответом служит последняя строка из табл. 7.2. Наконец, в четвертом примере содержится сразу три вопроса, которые можно интерпретировать так: «указать имена сущностей, их диапазоны частот и мощности

для всех носимых средств тактической радиосвязи». Компьютер должен ответить – ⟨ AN/PRC-88; 47-57; 0,3⟩ .

Таблицы, подобные 2.2, формально можно задать на множестве $H = I \times H_l \times \ldots \times H_k$, где $H_l$ – множества значений атрибута $j_l$, а I – множество имен сущностей, то есть в виде одинарных отношений $(l \le k)$. Такие отношения могут быть сведены к совокупности бинарных отношений. Тогда часть информации, содержащейся в табличной форме, может в явном виде не представляться.

Так, например, табл. 2.2 можно представить в виде графа, изображенного на рис. 2.17, на котором отражены в явном виде бинарные отношения R, имеющие место между сущностями и значениями атрибутов.

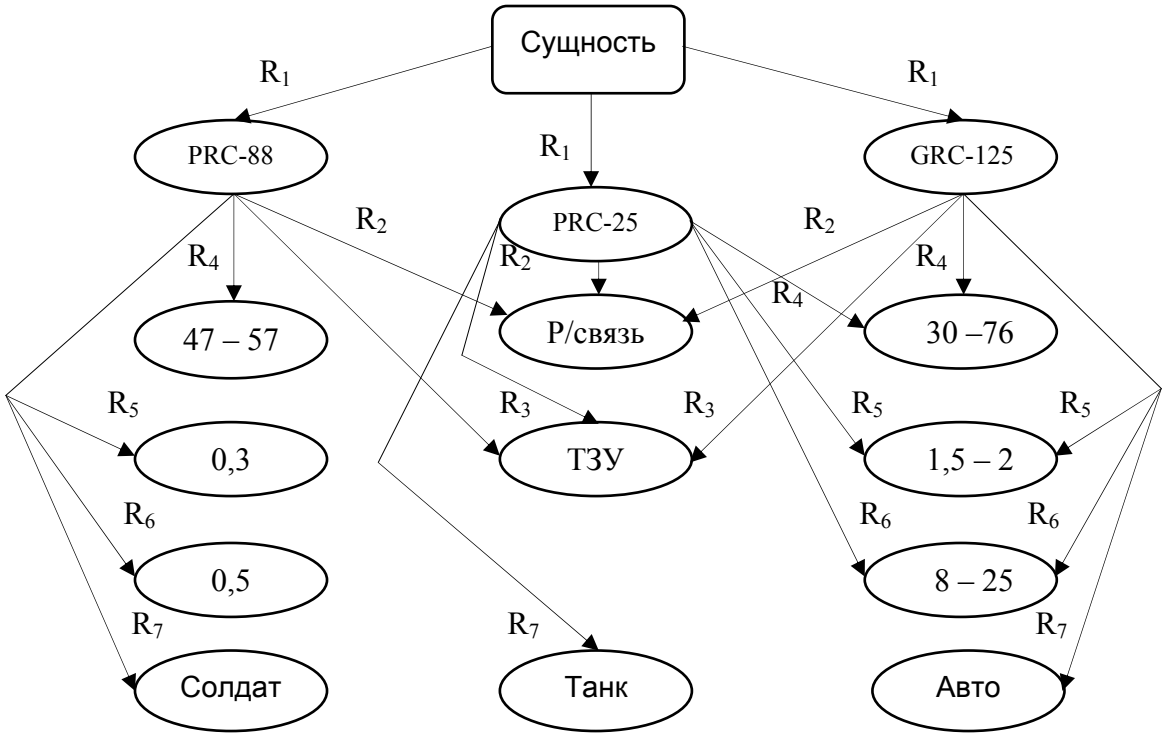


Рис. 2.17. *Граф бинарных связей, соответствующий данным табл. 2.2*

Сами атрибуты играют роль бинарных отношений, или дуг графа, а их значения – суть его вершины. В нашем случае отношение $R_1$ трактуется как «иметь имя», $R_2$ – «иметь назначение», $R_3$ – «принадлежать звену управления», $R_4$ – «иметь диапазон частот», $R_5$ – «иметь мощность», $R_6$ – «действовать на расстоя-

нии», $R_7$ – «перемещаться при помощи». Нетрудно видеть, что, несмотря на недостаточную наглядность, такая форма представления информации более компактна, чем табличная. Действительно, совпадающие значения атрибутов содержатся в графе только один раз.

На заре информатизации табличный язык с графовой формой представления информации составлял теоретическую основу разработки всевозможных баз данных – компьютерных программ, оперирующих с данными или фактами. Однако с переходом к созданию баз знаний выявилась его ограниченность. Напомним, что базой знаний в общем случае называется комплекс компьютерных программ, написанный на языке высокого уровня и включающий три основные составляющие:

- упорядоченные каким-либо способом факты и данные, отражающие модель профессиональной сферы (предметные знания);
- правила, позволяющие строить цепочки логических выводов и на этой основе делать обобщения и заключения, а так же вызывать определенные ассоциации (декларативные знания);
- управляющая или интерпретирующая структура, определяющая порядок и способы применения правил логического вывода для получения или трансформации информации (процедурные знания).

Границы между ними подвижны. Чем меньше знаний декларируется, тем больше предметных знаний необходимо, и наоборот. В реальных условиях различие между декларативными и процедурными знаниями не имеют существенного значения, поскольку определение того или иного правила может рассматриваться как декларативное знание. Базу данных следует считать частным случаем базы знаний.

Научные работы по изысканию эффективных методов структурного построения и программной реализации баз знаний стимулировали создание новых языковых средств типа RX-кодов,

семантических сетей и ролевых фреймов, обладающих более широкими возможностями по описанию фактов и данных. Кроме того, эти языки позволяют записывать и генерировать правила логического вывода (то есть, работать с декларативными знаниями), а также создавать управляющие структуры (то есть оперировать с процедурными знаниями).

***RX-коды.*** Базовыми единицами этого языка выступают отношения, которые обозначаются буквой $R_i$ с различными нижними индексами, и понятия, обозначаемые буквой $X_j^m$ с верхними и нижними индексами. В качестве правильных синтаксических выражений используются двойки $R_i\ X_j^m$, любые конъюнкции таких двоек и конъюнкции, получающиеся путем замены любого X в правильном синтаксическом выражении правильным выражением. Знак конъюнкции, если это не приводит к путанице, обычно опускается, а для указания зоны действия отношения используются круглые скобки. Эти правила и есть синтаксис языка RX-кодов.

Поясним сказанное. Пусть множество понятий состоит из $X_1$ и $X_2$, а множество отношений – их $R_1$ и $R_2$. Тогда синтаксически правильными будут, например, выражения $(R_1\ X_2)$, $(R_2\ X_2)(R_1\ X_2)(R_1\ X_1)$, а также выражение $((R_2\ X_2)(R_1\ (R_2\ X_1)(R_2\ X_2)))(R_1\ X_1)$, полученное из конъюнкции $(R_2\ X_2)(R_1\ X_2)(R_1\ X_1)$ заменой $X_2$ на $(R_2\ X_1)(R_2\ X_2)$.

Для интерпретации выражений языка RX-кодов всем понятиям и отношениям жестко предписываются некоторые семантические значения. В этом случае оказывается возможным производить вывод новых понятий через ранее определенные, которые для этого нового понятия выступают в качестве определяющих признаков.

Рассмотрим пример, иллюстрирующий такие свойства языка. Пусть $R_1$ есть отношение «быть элементом класса», $R_2$ – «использовать в качестве носителя» и $R_3$ – «включать в качестве элемента». Пусть также $X_1^0$ – «средство поражения самолетов», $X_2^0$ –

«самолет», $X_{10}^{2}$ – «ракета», $X_{11}^{1}$ – «средство наведения». Тогда запись: $X_{12}^{3} = (R_1\ X_1^{0})(R_2\ X_2^{0})(R_3\ X_{10}^{2})(R_3\ X_{11}^{1})$, определяет некоторое понятие, которое выражается следующим образом: «средство поражения самолетов, использующее в качестве носителя самолет и включающее в свой состав ракету и средство наведения», то есть $X_{12}^{3}$ – ракета «воздух-воздух».

Понятия $X_{10}^{2}$ и $X_{11}^{1}$ также можно задать в виде некоторых записей. Пусть $X_3^{0}$ – «средство доставки поражающих элементов», $X_4^{0}$ – «стабилизатор», $X_9^{1}$ – «двигатель, создающий тягу», $X_7^{0}$ – «пассивное средство наведения», $X_8^{0}$ – «электромагнитные излучения инфракрасного диапазона», $R_4$ – «принимать». Тогда понятие $X_{12}^{3}$ детализируется следующим образом: $X_{12}^{3} = (R_1\ X_1^{0})(R_2\ X_2^{0})(R_3\ ((R_1\ X_3^{0})(R_3\ X_4^{0})(R_3\ X_9^{1}))) (R_3\ ((R_1\ X_7^{0})(R_4\ X_8^{0})))$. Если, в свою очередь, детализировать понятие $X_9^{1}$, представив его в виде: $X_9^{1} = (R_1\ X_5^{0})(R_5\ X_6^{0})$, где $X_5^{0}$ – «силовая установка», $X_6^{0}$ – «принцип реактивного движения», а $R_5$ – «использовать», то окончательное определение понятия $X_{12}^{3}$ будет выглядеть так: $X_{12}^{3} = (R_1\ X_1^{0})(R_2\ X_2^{0})(R_3\ (R_1\ X_3^{0})(R_3\ X_4^{0})(R_3\ (R_1\ X_5^{0})(R_5\ X_6^{0})))(R_3\ ((R_1\ X_7^{0})(R_4\ X_8^{0})))$.

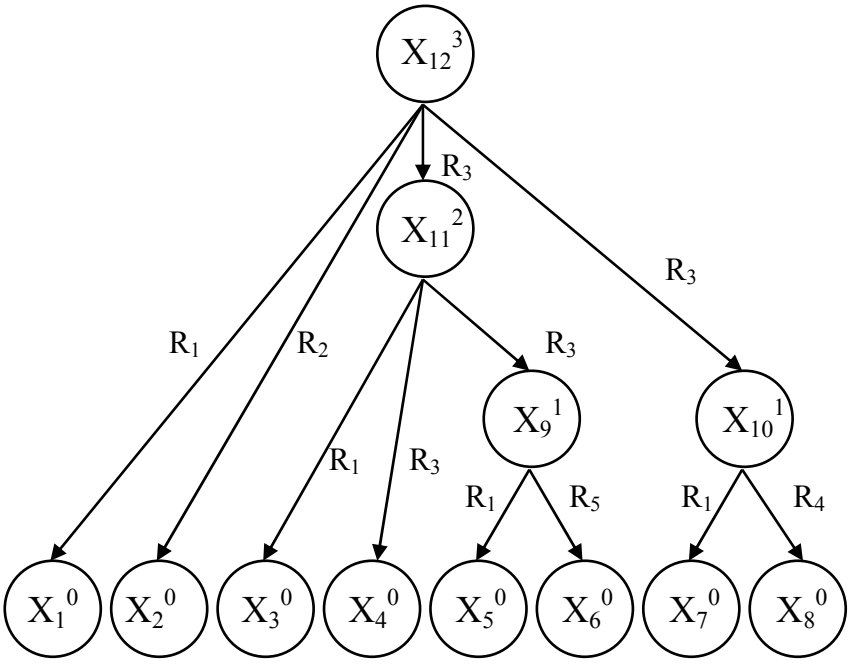


Рис. 2.18. *Граф, соответствующий понятию «ракета класса воздух-воздух»*

Для наглядности это выражение можно представить в виде графа, изображенного на рис 2.18, вершины которого соответствуют понятиям X, а дуги – отношениям R.

Естественная иерархия, присущая RX-кодам, позволяет хорошо описывать древовидные структуры, однако накладывает существенные ограничения на их описательные возможности. Дело в том, что определяемое в этом языке понятие всегда является корнем дерева, а отношения между понятиями одного уровня не учитываются. Для сочленения одноуровневых понятий требуется введение дополнительных (внеязыковых) средств. Вместе с тем именно это ограничение позволяет весьма эффективно использовать язык RX-кодов при организации баз данных, в которых родовидовые отношения играют определяющую роль. Для естественного языка древовидные конструкции не столь характерны. В нем отношения между понятиями имеют, как правило, сетевую структуру и содержат многочисленные циклы, в которых понятия определяются через самих себя. Поэтому при использовании RX-кодов для записи естественно-язычных текстов возникают определенные трудности. Фактически, RX-коды не могут адекватно отражать тексты, написанные на естественном языке.

***Семантические сети.*** Понятие семантической сети основано на очень простой идее о том, что память формируется через ассоциации (от лат. *associatio* – соединение) между понятиями. Понятие «ассоциативная память» появилось еще во времена Аристотеля и вошло в информатику в связи с работами по использованию простых ассоциаций для представления значения слов в базе данных [Элти Дж., Кумбс М., 1987]. Сегодня этот формализм всесторонне развит и широко используется для представления физических, управленческих, медицинских, военных и других видов знаний. Функциональным базовым элементом семантической сети служит тройная структура ($x_i$ r $x_j$), состоящая из двух «узлов» ($x_i$, $x_j$) и связывающей их «дуги» (r). Каждый узел представляет некоторое понятие, а дуга – отношение между парами понятий. Можно считать, что каждая такая тройка выражает простой факт. Например, дом ($x_1$) стоит на ($r_1$) горе ($x_2$). Дуга r имеет направленность, благодаря чему между понятиями x в рам-

ках определенного факта выражается отношение «субъект – объект». Кроме того, любой из узлов может быть соединен с любым числом других узлов, в результате чего существует возможность формировать сеть фактов.

Формально семантические сети задаются тремя классами термов: понятиями $X = \{x_1, x_2, \ldots, x_n\}$, именами $I = \{i_1, i_2, \ldots, I_m\}$ и отношениями $R = \{\rho, r_1, r_2, \ldots, r_k\}$. Отношение $\rho$ – особое. Оно означает «иметь имя», остальные отношения подразделяются на падежные, характеристические, причинно-следственные, иерархические, временные и топологические. Падежные отношения связывают предикат как основу действия, определяемого предложением, с остальными словами. Например, в предложении «вынуть мяч из корзины» слово «вынуть» – предикат. Им устанавливается предикатное отношение «вынимать» между двумя понятиями – «мяч» и «корзина». Характеристические отношения связывают характеристику с характеризуемым объектом и характеристику со значением характеристики. Например, в предложении «температура воздуха $5^0$C», характеризуемым объектом является «воздух», характеристикой – «температура», а значением характеристики – «$5^0$C». Они связаны характеристическими отношениями «быть характеристикой» и «иметь значение». Причинно-следственные отношения связывают понятия, одно из которых является причиной, а другое следствием. Эти отношения выражаются импликацией вида: $A \rightarrow B$, то есть: если A, то B. Иерархические отношения указывают на то, что один объект является составной частью другого объекта или – одно понятие определяется через другие понятия. Эти отношения выражаются словами «принадлежать к классу», «быть частью» и т.п. Временные отношения бывают двух типов: абсолютные и относительные. Абсолютные временные отношения устанавливаются между объектами и отрезками (точками) временной оси, а относительные – это связки «быть раньше», «одновременно», «быть позже» и другие. Топологические отношения связывают объекты с точками

какой-либо системы координат либо указывают на их взаимное расположение: «быть впереди», «быть сзади», «располагаться левее» и так далее.

Синтаксически правильными выражениями в языке семантических сетей считаются тройки $(x_i\ r_j\ x_l)$, $(x_i\ \rho\ i_j)$, $(r_i\ \rho\ i_j)$, где $x_i \in X$, $i_j \in I$, $\rho, r_i \in R$. Правильное выражение может быть образовано конъюнкцией (&) и дизъюнкцией ($\vee$) правильных выражений, а также операцией отрицания ($\neg$), которая применима как к элементам из R, так и к правильным выражениям. Кроме того, выражение, полученное заменой терма в правильном выражении правильным выражением, также является правильным.

Таблица 2.3

| | | |
|---|---|---|
| ПОНЯТИЯ | Эскадрилья | $x_1$ |
| | Мотопехотный полк | $x_2$ |
| | Механизированная бригада | $x_3$ |
| | Противник | $x_4$ |
| | Свои войска | $x_5$ |
| | Момент времени | $x_6$ |
| ИМЕНА | Первый | $i_1$ |
| | Второй | $i_2$ |
| | Третий | $i_3$ |
| | «Ч» | $i_4$ |
| ОТНОШЕНИЯ | Принадлежать к классу | $r_1$ |
| | Наносить удар | $r_2$ |
| | Овладевать позицией | $r_3$ |
| | Принадлежать к моменту времени | $r_4$ |
| | Следовать во времени за … | $r_5$ |

При разумном выборе обозначений с помощью семантических сетей можно выразить очень сложные совокупности фактов. В отличие от RX-кодов, семантические сети позволяют описывать не только постоянные отношения, присущие данному объекту, но и временные (ситуативные) отношения между объектами. Поясним сказанное на примере. Опишем на языке семантических сетей следующую ситуацию: «В момент времени «Ч» вторая эскадрилья наносит удар по позициям первой механизированной бригады противника. После удара третий мотопехотный полк овладевает позициями механизированной бригады противника». Введем обозначения термов, используемых в текстовом описании ситуации (табл. 2.3).

Тогда ситуация, описанная текстом на естественном языке, запишется на языке семантических сетей в виде следующего выражения:

$$(((x_1\ \rho\ i_2)(x_1\ r_1\ x_5))\ r_2\ ((x_3\ \rho\ i_1)(x_3\ r_1\ x_4)))\ r_4$$

$$(x_6\ \rho\ i_4)\ r_5\ (((x_2\ \rho\ i_2)(x_1\ r_1\ x_5))\ r_3\ ((x_3\ \rho\ i_1)(x_3\ r_1\ x_4))) \qquad (2.4)$$

Важная особенность языка семантических сетей заключается в его способности выводить новые понятия (обобщать ситуации) за счет выделения типовой структуры отношений.

Эта особенность не только используется для описания ситуаций – она оказалась весьма продуктивной при ситуационном поиске решений. Проиллюстрируем возможности языка семантических сетей по обобщению понятий на примере следующей ситуации: по дороге движутся один за другим несколько танков с номерами $i_1, i_2, \ldots i_{10}$.

Введем обозначения: $x_1$ – «боевая машина», $x_2$ – «тип боевой машины», $x_3$ – «дорога», $x_4$ – «направление», $x_0$ – «танк», $r_1$ – «обладать», $r_2$ – «двигаться по», $r_3$ – «находиться», $r_4$ – «быть сзади», $r_5$ – «находиться в ε-окрестности». Введем записи $(x_1\ \rho\ i_1)$; $(x_1\ \rho\ i_2)$; …;$(x_1\ \rho\ i_{10})$, которые фиксируют существование боевых машин с данными номерами. Для упрощения обозначим эти элементарные записи как $\alpha_1, \alpha_2, \ldots \alpha_{10}$. Теперь введем записи вида: $\beta_j : (\alpha_j\ r_1(x_2\ \rho\ i_{11}))$, где $j = 1, 2, \ldots 10$, смысл которых состоит в том, что боевая машина с номером j является танком.

Далее введем записи: $\gamma_j : (\beta_j\ r_3\ (x_3\ \rho\ \xi))$, означающие, что танк с номером i находится на дороге с именем $\xi$. В исходном тексте наименования дорог нет, но ясно, что все десять танков движутся по одной и той же дороге. Для фиксации этого факта и вводится свободная переменная $\xi$. Теперь введем записи: $\delta_j : (\gamma_j\ r_2\ (x_4\ \rho\ \eta))$, где $\eta$ играет для направления дороги такую же роль, что и имя $\xi$ для дороги, а смысл записи выражается фразой: все десять танков движутся по одной и той же дороге, в одном и том же направлении. Введем, наконец, следующие записи: $((\delta_1\ r_4\ \delta_2)\ \&\ (\delta_2\ r_4\ \delta_3)\ \&$

… & ($\delta_9$ $r_4$ $\delta_{10}$)) и (($\delta_1$ $r_5$ $\delta_2$) & ($\delta_2$ $r_5$ $\delta_3$) & … & ($\delta_9$ $r_5$ $\delta_{10}$)), смысл которых очевиден. Если их обозначить, как $\nu$ и $\pi$ соответственно, то выражение $\pi$ & $\nu$ есть полное описание ситуации, заданной текстом, приведенным в начале примера. Это же выражение трактуется как определение нового понятия $x_5 = \pi$ & $\nu$ – колонна танковой роты. Если же не фиксировать количество танков, а задать его в виде свободной переменной q и, кроме того, объявить свободными переменными имена танков, то запись для $x_5$ будет определять уже не колонну некоторой фиксированной танковой роты, а более общее понятие – танковая колонна.

Обобщение можно продолжить, заменив последовательно $x_2$, $x_1$ на свободные переменные, и получить таким образом конечную запись, трактуемую как колонна движущихся однородных объектов. Такое постепенное обобщение описаний в языке семантических сетей приводит, в конце концов, к построению так называемого минимального описания с максимальной свободой в заполнении переменных конкретными понятиями и со свободными параметрами типа переменной q в нашем примере. Это – самое важное свойство семантических сетей, позволяющее использовать их при создании иерархических баз знаний и разработке процедур поиска решений.

### 2.1.3. Ролевые языки

По мере развития логико-лингвистического моделирования выяснилось, что существует достаточно много ситуаций, которые не формализуются реляционными языками или требуют введения дополнительных конструкций, загромождающих компьютерные программы. Поэтому вслед за реляционными появились языки нового типа, названные ролевыми или фреймовыми [Minsky, 1975]. В этих языках понятия определяются совокупностью семантических (смысловых) ролей, которые выражают их сущ-

ность, и формально задаются строкой, называемой фреймом (от англ. *frame* – каркас, рамка):

$$\langle\, i;\, \rho_1,\, \rho_2,\ldots,\, \rho_m,\, \rho_{m+1},\, \rho_{m+2},\ldots,\, \rho_n \,\rangle, \qquad (2.5)$$

где i – имя понятия, $\rho_1$, $\rho_2$,…, $\rho_m$ – обязательные роли, а $\rho_{m+1}$, $\rho_{m+2}$,…, $\rho_n$ – необязательные роли. Множество необязательных ролей может быть пустым.

Общепринятого определения понятия «фрейм» нет. Большинство исследователей, следуя М. Минскому, исходят из того, что «*фрейм – это структура данных для представления стереотипной ситуации*».

Основное достоинство фреймов заключается в том, что с их помощью можно представлять информационные структуры произвольной сложности и достаточно адекватно отображать семантику моделируемых предметных областей. Кроме того, машинное представление знаний в виде фреймов подразумевает довольно полную и строгую их формализацию. Вместе с тем весьма трудно (даже для заранее известной предметной области) выделить сами фреймы. Какими по форме и структуре они должны быть? Сколько их нужно для адекватного описания ситуации? Как фреймы должны связываться друг с другом и передаются ли их свойства при сцеплении в сеть? Эти и другие вопросы пока не нашли исчерпывающих ответов и решаются различными авторами по-разному.

Рассмотрим в качестве примера фрейм, описывающий понятие «решение командира на бой». На естественном языке оно представляется записью: ⟨ Решение командира на бой; *замысел боевых действий*, *боевые задачи подчиненным*, порядок их взаимодействия, мероприятия по обеспечению, мероприятия по организации управления⟩. В приведенном примере выделенные роли являются обязательными в том смысле, что их определяет лично командир, а остальные необязательными – их планирует штаб с последующим докладом на утверждение командиру. Обязатель-

ные и необязательные роли, в свою очередь, являются сложными понятиями, представляемыми фреймами следующего уровня. Так, например, понятие «боевые задачи подчиненным» представляется следующим фреймом: ⟨ Боевые задачи подчиненным; подразделение, группировка противника, средства усиления, время перехода в атаку, направление, способ действий, срок доклада о готовности⟩ . Конкретной реализацией этого фрейма может служить, например, фрейм: ⟨ Боевая задача; 1-й мотострелковый батальон, опорный пункт 2-й механизированной бригады, минометный взвод 2-го мотострелкового батальона, 5.00 10.05, п. Сомово, атака с флангов после массированного минометного обстрела, 4.00 10.05⟩ .

Таким образом, ролями в каждом фрейме выступают другие фреймы, называемые слотами, что позволяет конструировать иерархические фреймовые описания рекурсивного вида:

$$\langle\, i;\ \rho_1(\langle\, i;\rho^1{}_1\langle\, i;\rho^2{}_1\langle\, \ldots\rho^M{}_1\rangle \ldots\rangle\,),\ \rho_2(\langle\, i;\rho^1{}_2\langle\ \ i;\rho^2{}_2\langle\, \ldots\rho^M{}_2\rangle \ldots\rangle\,),\ \ldots,$$

$$\rho_N(\langle\, i;\rho^1{}_N\langle\, i;\rho^2{}_N\langle\, \ldots\rho^M{}_N\rangle \ldots\rangle\,)\rangle\,. \qquad (2.6)$$

Существенным недостатком таких описаний является дублирование однотипной информации, что приводит при большем объеме данных к нерациональному расходу памяти. Поэтому в практических моделях, использующих фреймовые описания, применяется принцип наследования свойств. Суть его состоит в упорядочении рекурсии с помощью специального дерева (графа) зависимостей, в котором узлы соответствуют фреймам, а путь от корней к вершине указывает на порядок рекурсии. При этом вся информация, записанная во фреймах, лежащих на пути от корневой вершины до данной, автоматически переносится и в данную вершину.

Для представления знаний разрабатываются типовые фреймы, такие как: фрейм-состав, фрейм-соединение, фрейм-назначение, фрейм-параметр, фрейм-функция, фрейм-сценарий и другие. Для их компьютерного представления используются различные алго-

ритмические и программные средства: рекурсивные функции, KRL и FRL-языки, нормальные алгорифмы Маркова, исчисление λ-конверсий, язык универсального семантического кода (УСК) и другие.

Рассмотрим представление фреймов с помощью расширенной модификации λ-конверсий [Яцук, 1983]. В этом случае фрейм представляется в виде ориентированного графа, в котором помечены вершины и дуги. Одна из вершин выделена для предикатного (функционального) символа или числовой функции. Остальные вершины предназначены для аргументов выделенного символа. Аргументы находятся в некотором падежном отношении (метка на дуге) с выделенным символом. Для каждой вершины задано своя область допустимых значений, называемая сортом. На «аргументные места» фрейма допускается подстановка по определенным правилам (имеющим вид семантических ограничений) других фреймов, что позволяет представить взаимосвязанные структуры и вкладывать одно описание в другое, то есть слотом фрейма i-го уровня может быть фрейм i-1-го уровня и так далее, без ограничений.

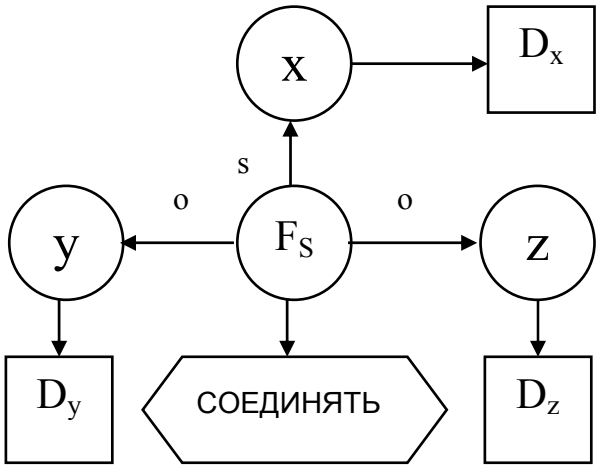

Рис. 2.19. *Образец фрейма-соединения*

Для примера возьмем фрейм-соединение ($F_S$), предназначенный для отображения различных типов соединений в технических системах. Образец такого фрейма показан на рис. 2.19. Он отображает ситуацию «субъект x соединяет объект y с объектом z» и описывается λ-выражением вида:

$$\lambda \{x: D_x, y: D_y, z: D_z. \text{СОЕДИНЯТЬ} (\langle s, x\rangle, \langle o, y\rangle, \langle o, z\rangle)\}, \quad (2.7)$$

где s и o – падежные отношения, соответственно субъект действия (то, что производит действие) и объект действия (то, над чем совершается действие), D – имя или сорт объекта (субъекта). Фреймы этого типа легко вкладываются друг в друга, что позволяет использовать их для описания иерархических структур в системах различного назначения.

Другой пример – фрейм-функция. Им описывается порядок расчета параметров $p_i$ некой системы при заданной функции

$$p_i(t) = f\,[(a, a_2, \ldots, a_N), t - \tau],$$

где $a_j$ – аргументы, к которым применяется функция, t – текущее время, $\tau$ – временная задержка.

Обобщенное $\lambda$-выражение для фреймов этого типа имеет вид:

$$\lambda\{p: D_p, t: D_t, f: D_f, a_1{:}D_1, \ldots, a_N: D_N.\ \text{ВЫЧИСЛИТЬ}\ (\langle res, p\rangle, \langle \tau, t\rangle) = (\langle vf, f\rangle)\ (\langle arg_1\ a_1\rangle, \ldots, \langle arg_N\ a_N\rangle)\}, \qquad (2.8)$$

где res – результат применения функции, arg – аргумент, vf – падежное отношение «вид функции», а запись x: $D_x$ означает, как и ранее, что переменная x имеет имя $D_x$.

При наличии стандартных вычислительных процедур использование подобных выражений позволяет задать алгоритм вычисления сложной составной функции произвольного вида.

Языковые средства мягких вычислений следует рассматривать в качестве дальнейшего развития и наращивания возможностей традиционных математических языков. Они предоставляют достаточно мощный инструментарий для формализованного описания реальных ситуаций, и одновременно открывают дорогу для внедрения в практику научных исследований современных компьютерных технологий. Более того, они сами оказывают позитивное влияние на развитие этих технологий, например, в части создания баз знаний, поиска решений, автоматизации управленческой деятельности в слабо формализуемых проблемных областях.

В функциях принадлежности нечетких множеств иногда усматривают вероятностную природу. Это – не так. Функция принадлежности в частном случае действительно может задаваться и интерпретироваться как вероятность. Однако, в общем случае – это понятийная или количественная характеристика, отражающая степень отношения какого-либо объекта к некоторой более общей области, например, с использованием шкал «ближе – дальше», «сильнее – слабее», «красивее – уродливее», «выше – ниже», «лучше – хуже» и др., а не только вероятностной шкалы «часто – редко».

## 2.2. Логико-лингвистические методы оценки и поиска решений

Традиционный подход к формализации задач оценки и поиска решений состоит в определении существенных характеристик проблемной ситуаций, которые совместно с существующими между ними отношениями могут быть описаны количественно. В терминах таких характеристик и отношений формируются ограничения и критерии выбора решений. При этом типовая задача оценки и поиска решений формально записывается в следующем виде:

$$\max (\min)\ F\ (x_1, x_2, \ldots, x_N) \tag{2.9}$$

при ограничениях

$$G_k\ (x_1, x_2, \ldots, x_N) = 0;\ k = 1, 2, \ldots, K; \tag{2.10}$$

$$x_i \in X;\ i = 1, 2, \ldots, N, \tag{2.11}$$

где переменные $(x_1, x_2, \ldots, x_N)$ – варьируемые характеристики задачи (компоненты решения); функционал (2.9) – критерий выбора решения (выбор max или min зависит от условия задачи и смысла критерия); соотношения (2.10) – уравнения связи между характеристиками (алгебраические, дифференциальные и др.); соотношения (2.11) – возможные значения варьируемых характеристик.

В моделях операционного типа для решения таких задач ис-

пользуется широкий арсенал методов математического программирования. В логико-лингвистических моделях использование этих методов неприемлемо по следующим причинам. Во-первых, в этих моделях переменные ($x_1$, $x_2$,…, $x_N$) не количественные, а лингвистические, то есть их значениями выступают не числа, а слова и предложения естественного или искусственного языка. Во-вторых, связи между переменными выражаются не в виде математических уравнений, а задаются с помощью лингвистических, логических или словесных выражений. В-третьих, критерии выбора формулируются не в виде точного математического функционала, а описываются качественными формулировками, например в виде текстовых указаний по предпочтительности, недопустимости или желательности того или иного варианта решения.

Поэтому в логико-лингвистических моделях реализуются специфические методы оценки и поиска решений, ориентированные на качественное (понятийное) описание компонентов решений, связей между ними и критериев выбора. В настоящее время в теории логико-лингвистического моделирования еще не созданы универсальные методы поиска решений, в полной мере обеспечивающие практические нужды системных исследований. Поэтому мы ограничимся рассмотрением трех практических примеров.

### 2.2.1. Поиск решений на семантических сетях

Существо подхода к поиску решений, реализуемого этим методом, состоит в следующем. Для рассматриваемой проблемной области выделяется группа качественных характеристик, существенных с точки зрения принятия решений, и перечисляются их возможные лингвистические значения. Используя эти характеристики, формируют те или иные высказывания, описывающие проблемную ситуацию. Среди таких высказываний могут оказаться как допустимые, так и недопустимые. К недопустимым относятся высказывания либо не имеющие смысла в данной проблемной

области, либо очевидно нецелесообразные для конкретной ситуации. Все остальные высказывания считаются допустимыми и образуют в совокупности базу предметных знаний.

Модель предметных знаний формализуется с помощью семантической сети, вершинами которой являются имена характеристик, а дугами – их возможные значения. Формулировка задачи поиска решения на такой сети осуществляется путем фиксации значений характеристик, описывающих условия, в которых принимается решение, и указания характеристик, значения которых необходимо определить. Искомые значения характеристик определяются в два этапа.

Вначале выявляются допустимые решения, то есть такие решения, характеристики которых удовлетворяют заданным условиям. Затем (если решение неоднозначное) с помощью некоторой системы предпочтений отыскивается рациональное решение из числа допустимых. Система предпочтений образует базу процедурных знаний, которая позволяет с использованием одной и той же семантической сети решать самые различные задачи поиска решений в данной проблемной области, в том числе и взаимосвязанные. Иными словами, реализуется классическая двухэтапная схема принятия системного решения, когда вначале анализируется обстановка и указывается, что можно и чего нельзя делать, а затем (если это возможно) определяется, что лучше делать.

Поясним сказанное на упрощенном примере описания ситуации из военной области. «Зеленые» осуществляют высадку воздушного десанта для захвата и уничтожения тылового объекта «синих». «Синие», занимая круговую оборону объекта, создали подвижную оперативную и резервную группы, которые могут применяться как для уничтожения десанта, так и для сковывания его действий до подхода сил старшего начальника путем атаки с ходу или путем занятия обороны. Боевые действия могут вестись на маршруте выдвижения десанта после сбора и приведения его подразделений в боевую готовность, а также в районе высадки

десанта до и после приведения его подразделений в боевую готовность. «Синие» получили сигнал оповещения о высадке воздушного десанта «зеленых» и приказ на начало боевых действий. Войска «синих», находящиеся в данном районе, приведены в полную боевую готовность. Личный состав занял оборонительные позиции и посты по боевому расписанию. Подвижная и резервная группы выдвинуты в угрожаемые районы. Командование «синих» уточняет оперативную обстановку и должно принять решение на боевые действия.

Введем характеристики, описывающие за «синих» рассматриваемую ситуацию, и их возможные (для данного примера) значения:

- источники действий ($X_1$): подвижная оперативная группа – $a_1$; резервная группа – $a_2$;
- цели действий ($X_2$): сковывание сил десанта – $b_1$; уничтожение десанта– $b_2$;
- способы действий ($X_3$): оборона – $c_1$; атака с ходу – $c_2$;
- место совершения действий ($X_4$): на маршруте выдвижения десанта – $d_1$; в районе высадки десанта – $d_2$;
- время совершения действий ($X_5$): до приведения десанта в боевую готовность – $e_1$; после приведения десанта в боевую готовность – $e_2$.

Таким образом, для описания рассматриваемой ситуации вводится пять лингвистических переменных $X_1$, $X_2$, $X_3$, $X_4$, $X_5$, каждая из которых может принимать по два значения. Естественно, что для полного описания реальной ситуации этих данных недостаточно, однако, учитывая иллюстративный характер примера, мы ограничимся таким минимальным объемом информации. Лингвистические уравнения связи между введенными переменными, соответствующие соотношениям (2.10) и определяющие допустимые комбинации значений переменных, будем формировать, задавая невозможность или нецелесообразность сочетаний их

значений. При этом все другие комбинации будем считать допустимыми.

Предположим, что в нашем примере невозможными (нецелесообразными) являются следующие сочетания значений: $X_4 = d_1$, $X_5 = e_1$ (боевые действия на маршруте выдвижения десанта до момента сбора и приведения его подразделений в боевую готовность); $X_2 = b_2$, $X_3 = c_1$ (уничтожение десанта путем ведения обороны); $X_1 = a_1$, $X_2 = b_2$ (использование резервной группы для сковывания сил десанта). На практике формирование таких условий осуществляется исходя из результатов оценки (уточнения) складывающейся оперативной обстановки (за себя и за противника) и на основе указаний, поступивших от старшего начальника. Кроме того, используются нормативные данные по боевым возможностям сил и тактико-техническим характеристикам применяемого вооружения.

Допустимые сочетания переменных $X_1$, $X_2$, $X_3$, $X_4$, $X_5$ представлены в виде семантической сети на рис. 2.20.

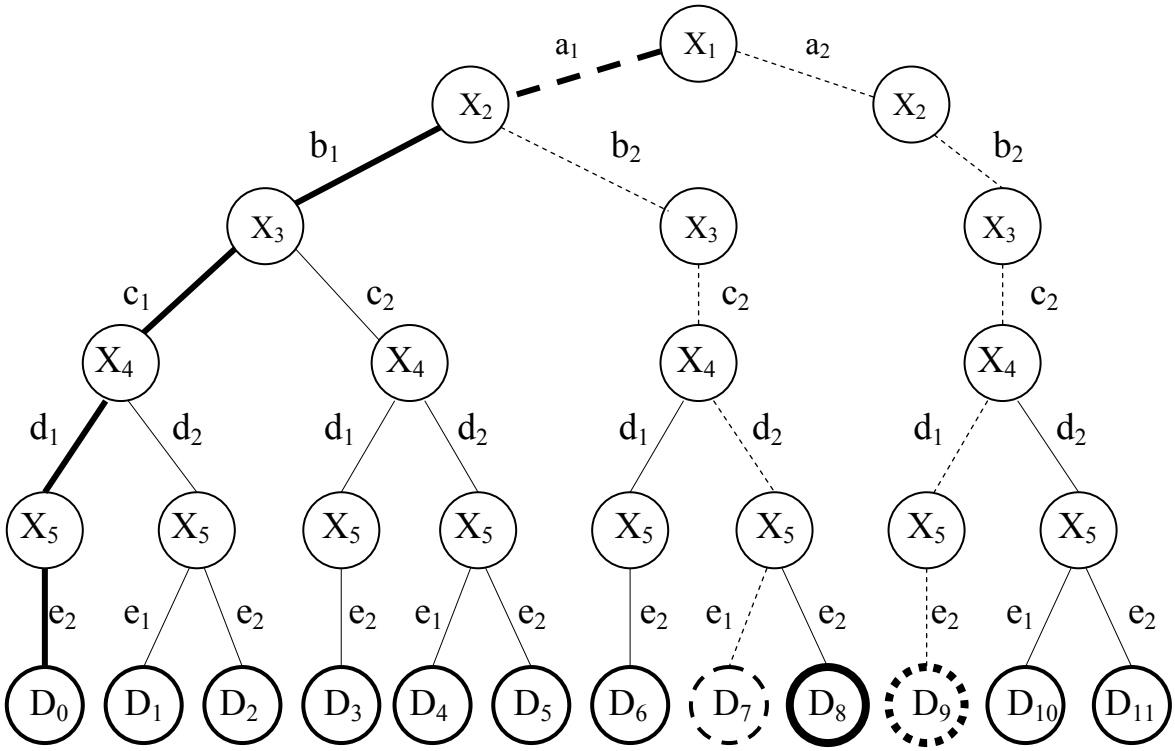


Рис. 2.20. *Семантическая сеть поиска решений (пример)*

Каждой цепочке $D_X$,(x = 1,2,...,11) этой сети можно поставить

в соответствие некоторое осмысленное в рассматриваемой ситуации высказывание, связанное с физически реализуемым действием. Например, цепочке $D_0 = (a_1, b_1, c_1, d_1, e_2)$, отмеченной на рисунке жирной линией, соответствует следующее высказывание: подвижная оперативная группа ($a_1$) с целью сковывания сил десанта ($b_1$) занимает оборону ($c_1$) на маршруте выдвижения десанта ($d_1$) после приведения подразделений десанта в боевую готовность ($e_2$). Здесь понятия «с целью», «занимает» являются вспомогательными и представляют собой фактически имена лингвистических переменных, выраженные в соответствии с грамматикой русского языка. Их использование придает фразе соответствующую смысловую окраску.

Фиксируя определенные значения некоторых переменных $X_1$, $X_2$, $X_3$, $X_4$, $X_5$, то есть задавая «вход» ситуации принятия решения, можно определить ее «выходы» путем вычисления значений некоторых или всех остальных переменных с помощью введенных выше отношений недопустимости сочетаний значений переменных. При этом процедура поиска решения формально сводится к фиксации определенных звеньев семантической сети и нахождению всех допустимых цепочек, проходящих через эти звенья. Далее введем какой-либо критерий предпочтительности и с его использованием из всех допустимых цепочек выберем одну, соответствующую рациональному решению.

Сформулируем теперь одну из возможных задач поиска решений в рассматриваемой ситуации.

*Задача* 1. Определить действия «синих» ($D^1_x$), которые они должны осуществить для уничтожения воздушного десанта «зеленых», если известно, что «зеленые» уже завершили высадку десанта, их подразделения собрались в районе сосредоточения и готовы к боевым действиям.

В формализованном виде такая задача формулируется так: необходимо определить такие цепочки $D^1_x = (X_1, X_2, X_3, X_4, X_5)$ семантической сети (рис. 8.1), для которых $X_2 = b_2$ и $X_5 = e_2$. Здесь

переменные $X_2$, $X_5$ – «входы», а переменные $X_1$, $X_4$, $X_4$ – «выходы» ситуации поиска решений. Как видно из рис. 10.1, существует четыре таких цепочки, то есть четыре допустимых решения: $D^1_6 = (a_1, b_2, c_2, d_1, e_2)$; $D^1_8 = (a_1, b_2, c_2, d_2, e_2)$; $D^1_9 = (a_2, b_2, c_2, d_1, e_2)$; $D^1_{11} = (a_2, b_2, c_2, d_2, e_2)$.

Для выявления из найденных допустимых решений рационального варианта необходимо сформулировать соответствующий критерий выбора. Введем следующий критерий: лучшим (наиболее рациональным) из числа допустимых считается такое решение $D_x$, которое ближе всех находится к некоторому, заранее сформулированному, эталонному решению E. За меру близости между решениями $D_x$ и E примем величину $\rho (D_x, E)$, равную числу несовпадающих значений одноименных компонентов $X_i$ (возможно, взвешенных некоторыми весами $\alpha_i \geq 0$, $\sum \alpha_i = 1$). Нетрудно проверить, что такая мера удовлетворяет всем аксиомам метрики и имеет простой физический смысл: чем меньше несовпадений значений одноименных компонентов решений, тем меньше отличаются эти решения. Формально можно записать:

$$D_{opt} = \text{Arg } \min_{x \in X} [\rho (D_x, E)], \qquad (2.12)$$

где X – номера допустимых по условию задачи цепочек семантической сети (в нашем примере X = 6, 8, 9, 11).

Эталонные решения E формулируются исходя из: а) имеющегося опыта решения подобных задач на учениях, на тренировках и в ходе предшествующих боевых действий; б) в соответствии с указаниями, содержащимися в решении старшего начальника; в) на основе использования общих положений различных регламентирующих документов (уставов, наставлений, руководств и т.п.). Возможны иные пути определения эталонных решений и использование других критериев выбора рациональных решений. Для рассматриваемого метода это не принципиально. Важен принцип: в бою не бывает тупиковых ситуаций, если ситуация назрела, то решение должно быть принято. Карл Клаузевиц говорил: «*На вой-*

*не большей частью дело заключается не в том, чтобы решаться на лучшее, а хоть на что-нибудь, лишь бы это что-нибудь было энергически приведено в исполнение*» [Макаров, 1942].

В том случае, когда не представляется возможным указать эталонные решения, на базе той же семантической сети можно построить другую схему поиска решений, позволяющую применительно к конкретной ситуации последовательно ответить на вопросы типа: что категорически запрещено делать, чего не рекомендуется делать, чего лучше не делать и что надо обязательно делать. В этом случае вместо эталонных решений вводятся запрещающие, предупреждающие, рекомендующие и предписывающие правила, и производится соответствующая «раскраска» дуг сети.

Предположим теперь, что в качестве эталонного из каких-либо соображений выбрано следующее утверждение: «наибольший успех в разгроме воздушного десанта противника достигается при атаке с ходу силами подвижных оперативных групп в период выброски, сбора и приведения его подразделений в боевую готовность». Для рассматриваемой задачи I это означает, что эталонное решение $E^1 = D^7 = (a_1, b_2, c_2, d_2, e_1)$. На рис. 2.20 цепочка, соответствующая этому эталонному решению, показана штрихованной линией. Расстояние каждого из допустимых решений $D^1_6$, $D^1_8$, $D^1_9$, $D^1_{11}$ до рационального будет $\rho (D^1_6, E^1) = 2$, $\rho (D^1_8, E^1) = 1$, $\rho (D^1_9, E^1) = 3$ и $\rho (D^1_{11}, E^1) = 2$. Следовательно, решение $D^1_8$ меньше всех отличается от эталона, и в соответствии с введенным выше критерием, является более предпочтительным. Суть его состоит в том, что наиболее рациональными действиями «синих» при разгроме воздушного десанта «зеленых» после сбора и приведения его подразделений в боевую готовность является атака с ходу силами подвижной оперативной группы в районе десантирования.

Отметим некоторые особенности поиска решений на семантических сетях, когда рассматриваются не отдельные задачи, а

совокупности взаимосвязанных задач принятия решений. В практике – это наиболее типичный случай. Для простоты рассмотрим решение двух взаимосвязанных задач, полагая, что аналогичные рассуждения справедливы при решении k (k >2) взаимосвязанных задач. Пусть одной из таких задач является задача 1, а другой – *задача* 2, полностью совпадающая с ней по условию и отличающаяся лишь эталонным решением. Так как для описания этих задач используется одна и та же семантическая сеть (рис. 2.20), то условия второй задачи могут быть формализованы аналогично тому, как это делалось в первой задаче, но с заменой переменных X на Y. А именно, необходимо определить все такие цепочки $D^2_y = (Y_1, Y_2, Y_3, Y_4, Y_5)$ семантической сети, для которых $Y_2 = b_2$, $Y_5 = e_2$. В качестве эталона для задачи 2 выберем решение $E_2 = (a_2, b_2, c_2, d_1, e_2)$. В этом случае решение задачи очевидно. Согласно введенному критерию, рациональным следует признать решение $D^2_9$, поскольку оно полностью совпадает с эталонным – $\rho (D^2_9, E_2) = 0$.

Предположим теперь, что боевая обстановка складывается так, что у «синих» для разгрома десанта «зеленых» имеется возможность привлечь силы какой-либо одной группы – подвижной оперативной или резервной. Это означает, что решения задач 1 и 2 связаны таким образом, что: $X_1 = Y_1$, то есть источники действий при реализации этих задач должны быть одними и теми же. Заметим, что полученные решения задач 1 и 2 не удовлетворяют условию $X_1 = Y_1$, так как значение первой компоненты решения $D^1_8 = a_1$, а решения $D^2_9 = b_2$, то есть при решении задачи 1 рациональным было признано применение сил подвижной оперативной группы, а задачи 2 – сил резервной группы. Следовательно, если учитывать условие $X_1 = Y_1$, то полученные ранее решения не являются совместно рациональными, и требуется поиск некого компромисса, то есть необходимо произвести координацию этих решений.

Координацию решений задач 1 и 2 можно осуществить путем

регулирования связи $X_1 = Y_1$. При этом под рациональным регулированием связи будем понимать выбор такой из ее допустимых реализаций, которая обеспечивает минимум (по реализациям) максимума (по задачам) отклонений рационального в каждой задаче решения от своего эталона, или формально:

$$W = \min_{X1=Y1} \max[\min_{x \in X} \rho (D^1_x, E^1), \min_{y \in Y} \rho (D^2_y, E^2)]. \quad (2.13)$$

Как видно из приведенной записи, такой критерий позволяет найти компромиссное решение, которое в нашем случае интерпретируется следующим образом. Поскольку рациональные решения каждой задачи в отдельности достигаются при прямо противоположных стратегиях ($X_1 \neq Y_1$), то максимизация наименьших отклонений решения каждой задачи от своего эталона обеспечивает наибольшее приближение каждой отдельной задачи к своему рациональному решению. Минимизация же по стратегиям наибольших отклонений позволяет выбрать одну компромиссную стратегию, лучшую для двух задач в целом. В рассматриваемом примере возможны две стратегии: $X_1 = Y_1 = a_1$ и $X_1 = Y_1 = a_2$.

Для первой стратегии существует два допустимых решения $D_6$ и $D_8$, одно из которых $D_8$ – рациональное для задачи 1 и оба – нерациональные для задачи 2, причем $\rho_1 (D_6, E^1) = 2$, $\rho_1 (D_8, E^1) = 1$, $\rho_2 (D_6, E^2) = 1$, $\rho_2 (D_8, E^2) = 2$, $\max (\rho_1, \rho_2) = 2$. Для второй стратегии тоже существуют два допустимых решения $D_9$ и $D_{11}$, где $D_9$ – решение, рациональное для задачи 2 и оба – нерациональные для задачи 1, причем $\rho_1 (D_9, E^1) = 3$, $\rho_1 (D_{11}, E^1) = 2$, $\rho_2 (D_9, E^2) = 2$, $\rho_2 (D_{11}, E^2) = 1$, $\max (\rho_1, \rho_2) = 3$. Таким образом, в соответствии с введенным минимаксным критерием рациональным при условии $X_1 = Y_1$ следует признать первый вариант реализации связи ($X_1 = Y_1 = a_1$) с окончательным комплексным решением $D_8 = (a_1, b_2, c_2, d_2, e_2)$.

Заметим, что если без учета связи $X_1 = Y_1$ отклонения рациональных решений от эталонов в задачах 1 и 2 составляли соответственно $\rho_1 = 1$, $\rho_2 = 0$, то в результате регулирования связи эти от-

клонения будут соответственно $\rho_1 = 1$, $\rho_2 = 2$. Следовательно, в данном случае координирование реализовано путем увеличения показателя $\rho_2$ в задаче 2, для которого он при раздельном решении был наименьшим. Именно в этом состоит сущность введенного критерия координации: проявление осторожности в оценке возможных ситуаций и в сбалансированном учете интересов участников коалиции (в нашем случае задач). На практике (помимо рассмотренного минимаксного критерия) используются другие критерии координации. В частности, возможно применение так называемого усредняющего критерия, по которому в качестве ра-

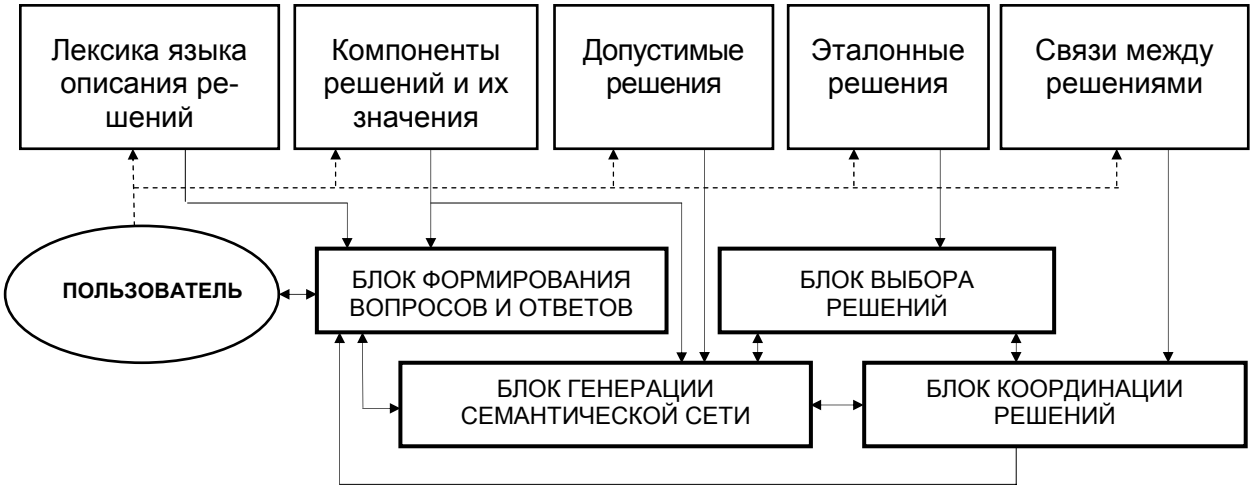


Рис. 2.21. *Функциональная схема процедуры поиска решений на семантических сетях*

циональной реализации связи выбирается значение, обеспечивающее минимизацию среднего (по задачам) отклонения от эталонов. В отличие от минимаксного критерия, использование усредняющего критерия связано с известным риском: плохое текущее управление может быть оценено как хорошее в среднем.

Итак, мы рассмотрели существо метода поиска решений на семантических сетях. При этом для простоты анализировалась сильно упрощенная ситуация, в которой оператор средней квалификации без привлечения методического аппарата мог бы с успехом отыскать наиболее рациональное решение. В реальных же ситуациях, характеризующихся числом компонентов решений порядка 10-15 и количеством их возможных значений около 40-

50, даже высококвалифицированный оператор испытывает определенные затруднения, особенно в условиях жесткого лимита времени. На помощь ему должен прийти компьютер.

Функциональная схема компьютерной процедуры поиска решений на семантических сетях представлена на рис. 2.21.

В соответствии с этой схемой, исходными данными для поиска решений являются: наименования компонентов решений, определяющих вершины семантической сети; возможные значения компонентов решений и правила формирования допустимых комбинаций этих значений, определяющие дуги семантической сети; лексемы естественного языка описания решений; эталонные решения; связи между решениями. Операционная часть процедуры состоит из блоков формирования запроса и ответа, генерации семантической сети, выбора решений и координации решений. Блок формирования запроса и ответа реализует диалог пользователя с компьютером на естественном профессионально ориентированном языке. Кроме того, здесь же происходит проверка запроса (постановки задачи) на корректность, то есть осмысленность в рамках принятой формализации, анализ и уточнение запроса, пополнение и модификация формальных грамматик.

В блоке генерации семантической сети в соответствии с заложенными формальными грамматиками производится порождение альтернатив решения, то есть формирование цепочек, удовлетворяющим условиям задачи. При этом с целью экономии памяти и ускорения поиска формируются цепочки, относящиеся только к данной задаче. Полученные допустимые решения поступают в блок выбора решений, где среди них отыскиваются наиболее близкие к эталонным. Далее эти решения в кодах поступают в блок формирования ответа, где производится их перевод на естественный язык. При рассмотрении взаимосвязанных задач результаты решения каждой из них поступают в блок координации, в котором выполняются операции по регулированию связей, например, согласно рассмотренному минимаксному критерию.

Метод поиска решений на семантических сетях, реализованный в виде компьютерных программ, позволяет решать достаточно широкий класс слабо формализуемых задач. Его реализация связана с разработкой естественного проблемно ориентированного языка, а также с созданием декларативных и предметных баз знаний в рассматриваемой проблемной области. Другой важной особенностью этого метода является присутствие пользователя (исследователя, оператора) как на этапе обучения (пополнение модели знаний, уточнение компонентов решений, ввод эталонных решений и т.п.), так и на этапе решения конкретных задач (передача управления очередному блоку, изменение эталонных решений, уточнение постановок задач и т.п.).

### 2.2.2. Ситуационный поиск решений

***Идея.*** Ситуационный поиск решений применяется при следующих условиях. Все сведения о состоянии управляемого объекта, целях управления и множестве возможных решений по управлению сообщаются объекту, принимающему решение, в виде последовательности фраз на естественном языке. Ставится задача поиска не оптимальных, а рациональных решений, то есть решений, удовлетворяющих некоторым неформальным правилам, условиям и ограничениям. Количество возможных ситуаций, требующих принятия решений, намного превосходит количество возможных решений.

В своей основе ситуационный поиск решений представляет собой удачную попытку реализовать в компьютерной форме рефлексные механизмы мышления, свойственные психическому комплексу человека. Идея метода довольно проста и иллюстрируется схемой, приведенной на рис. 2.22. Пусть в каждый текущий момент времени мы можем получить неформальное описание складывающейся ситуации, отражающей: состояние объекта управления $\{S_1\}$; состояние управляющего объекта $\{S_2\}$ и текущие цели управления $\{C\}$. Обозначим это описание через $S(t)$.

Предполагается, что информации, содержащейся в S(t), вместе с той информацией, которая уже имеется в памяти управляющего объекта, достаточно для принятия решения по управлению, которое формируется из заданного множества элементарных решений $\{p_i\}$. Решение при этом рассматривается как некоторая цепочка, состоящая из элементарных решений, или дерево, путь по которому определяется множеством возникающих при управлении ситуаций на входе управляющего объекта. Тогда для каждого момента времени t можно рассмотреть элементарную задачу определения того решения $P \in \{p_i\}$, которое необходимо принять в момент времени t при наличии ситуации S(t). Это означает, что в процессе обучения необходимо на множестве {S(t)} выделить такие подмножества-классы, каждому из которых соответствовало бы некоторое элементарное решение $P \in \{p_i\}$.

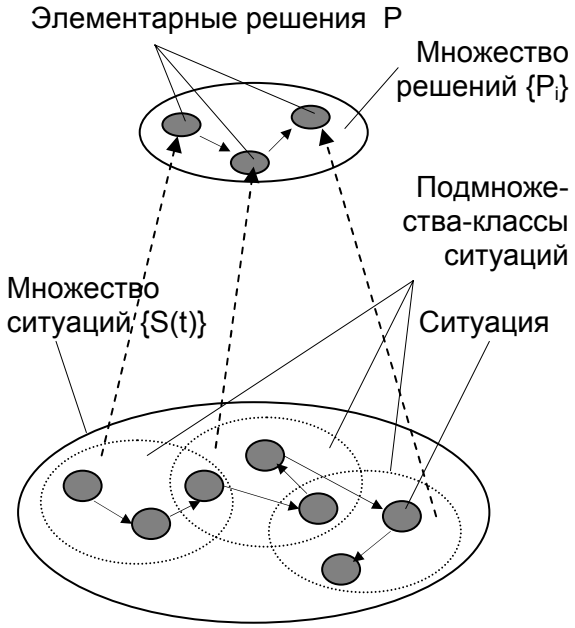

Рис. 2.22. *Идея ситуационного поиска решений*

Отметим две особенности такой постановки задачи. Во-первых, поскольку мы имеем дело с неформальным описанием ситуаций, то не представляется возможным четко выделить подмножества-классы в {S(t)}, соответствующие элементарным решениям. В реальных задачах границы между подмножествами-классами размыты и, фактически, можно говорить не о разбиении множества {S(t)}, а о его покрытии. Во-вторых, учитывая, что в практических задачах мощность множества {S(t)} намного превышает мощность множества $\{p_i\}$, обобщение текущих ситуаций в подмножества-классы следует проводить постепенно, а полезность каждого промежуточного обобщения должна проверяться путем сравнения с экспертными решениями по этому вопросу.

В реализации ситуационного поиска решения можно выделить следующие основные стадии: описание ситуаций, обобщение ситуаций и выбор решения.

***Описание ситуаций***. Для описания ситуаций используется модификация языка семантических сетей, названная языком ситуационного управления [Поспелов, 1975]. Его лексику образуют: базовые понятия {v}; базовые отношения {r}; имена (i); элементарные решения {p} и оценки {O}. Из элементов этих множеств по определенным правилам грамматики (подобным правилам, рассмотренным в разделе 8.3) конструируются тексты, дающие описание входных ситуаций S(t). Множество базовых понятий состоит из физических понятий и понятий-классов. Физические понятия задаются набором признаков, значения которых определяют эти понятия (например, совокупностью показаний некоторых приборов, установленных на объекте управления). Будем обозначать этот набор как$\{\pi_1, \pi_2, \ldots, \pi_n\}$. Понятия-классы есть просто совокупности физических понятий, однородные с некоторой точки зрения, например, система, комплекс, средство, элемент и т.п. Отнесение того или иного физического понятия к понятию-классу является для модели внешней функцией, и правила этого отнесения в модели не формируются. Имена служат для персонификации элементов, входящих в некоторое понятие-класс. Отношения r устанавливают связи между элементами множеств {π}, {v}, {i}, {p}. Таким образом, описание ситуации с использованием такого языка представляет собой некоторую структуру, составленную из элементов множеств {π}, {v}, {i}, {p}, связанных отношениями {r}. Возможности этого языка продемонстрируем на упрощенном примере описания следующей ситуации. Десятый мотострелковый батальон (мсб) «зеленых» занимает оборону населенного пункта «О». Пятой механизированной бригаде (мбр) «синих», усиленной вертолетами огневой поддержки, поставлена задача захватить населенный пункт «О». Первый мотострелковый (мсб)

и второй танковый (тб) батальоны пятой мбр «синих» находятся на марше и приближаются к зонам действий минометных и противотанковых средств «зеленых». Вертолеты огневой поддержки «синих» вошли в зону действия зенитных ракет «зеленых».

Введем обозначения. Понятия-классы: $v_1$ – мсб, $v_2$ – «синие», $v_3$ – населенный пункт, $v_4$ – мбр, $v_5$ – «зеленые», $v_6$ – вертолеты огневой поддержки, $v_7$ – тб, $v_8$ – марш (движение), $v_9$ – зона действия противотанковых средств, $v_{10}$ – зона действия зенитных ракет, $v_{11}$ – зона действия минометных средств, $v_{12}$ – мпб. Имена: $i_1$ – «10», $i_2$ – «О», $i_3$ – «5», $i_4$ – «1», $i_5$ – «2». Отношения: $r_1$ – «иметь имя», $r_2$ –«находиться на (в)», $r_3$ – «быть средством усиления», $r_4$ – «приближаться к», $r_5$ – «принадлежать к классу», $r_б$ – «одновременно», $r_7$ – «оборонять», $r_8$ – «захватывать». Тогда с учетом введенных обозначений рассматриваемая ситуация может быть описана выражением: $S(t) = ((v_1\ r_5\ v_5)(v_1\ r_1\ i_1)(v_1\ r_7\ v_3\ r_1\ i_2))$ & $((v_4\ r_5\ v_2)(v_4\ r_1\ i_3)(v_4\ r_8\ v_3\ r_1\ i_2)\ (v_6\ r_3\ v_4))$ & $(((v_{12}\ r_5\ v_2)(v_{12}\ r_1\ i_4)(v_{12}\ r_4\ v_{10})(v_{12}\ r_2\ v_8))\ r_6\ ((v_7\ r_5\ v_2)(v_7\ r_1\ i_5)\ (v_7\ r_4\ v_9)(v_7\ r_2\ v_8)))\ r_6\ (v_6\ r_2\ v_{11})$.

Использование подобных структур исключает неоднозначность понимания описываемых ситуаций (характерное для естественного языка). Они «понятны» компьютеру, то есть могут легко записываться, храниться и обрабатываться, образуя базу предметных знаний. Такие тексты допускают формальные преобразования своих конструкций, в соответствии с грамматикой данного языка, что позволяет выводить новые понятия и обобщать ситуации (разделять множество ситуаций на подмножества-классы).

***Обобщение ситуаций***. При наличии обозримого множества ситуаций $\{S(t)\}$ можно было бы поставить каждой из них соответствующее решение. Однако в практически интересных случаях мощность множества $\{S(t)\}$ настолько велика, что нет никакой надежды на сопоставление «ситуация – решение». Одним из способов уменьшения количества ситуаций является их обобщение. Именно поэтому проблема обобщения ситуаций является центральной в ситуационном поиске решений.

В настоящее время практическое применение нашли три способа обобщения ситуаций. Самым простым из них является обобщение по именам: к одному классу ситуаций относятся все ситуации, описание которых отличается лишь именами, стоящими на определенных местах, и которые требуют при своем возникновении однотипных решений. Вторым способом обобщения ситуаций является формирование функций принадлежности к классу ситуаций на основе значений признаков $\{\pi\}$, которые рассматриваются как лингвистические переменные с соответствующими функциями принадлежности. С использованием операций над функциями принадлежности строятся нечеткие алгоритмы распознавания типов ситуаций. Третьим способом является обобщение ситуаций по структуре отношений, которыми они описываются. Суть этого способа сводится к выделению в формализованных описаниях ситуаций, некоторых типовых структур (фреймов), которыми и определяются обобщенные ситуации. Пример такого способа формирования обобщенных ситуаций (понятий) рассматривался выше при описании языка семантических сетей. Следует отметить, что в ситуационном поиске решений именно этот способ обобщения оказался наиболее продуктивным.

В общем случае обобщение ситуаций можно представить в виде многослойной структуры, между слоями которой имеются связи, указывающие переход от более мелких описаний к более крупным. Эти связи устанавливаются согласно изложенным способами обобщения и взвешиваются оценками Oj, значения которых в процессе обучения модели вводятся пользователем.

***Выбор решений*** в ситуационных моделях осуществляется с помощью специальных корреляционных правил (или правил вывода), имеющих в простейшем случае следующий вид: $S^1(t) \Rightarrow p_1$. Смысл этого правила состоит в том, что при наличии ситуации $S^1(t)$ необходимо принять решение $p_1$. В частности, для рассмотренного выше примера можно ввести правило: $(v_6\ r_2\ v_{11}) \Rightarrow p_1$, ко-

торое означает, что при наличии в описании исходной ситуации структуры, стоящей в левой части правила или ее эквивалента (с точностью до формальных преобразований, допускаемых грамматикой данного языка), необходимо принять решение $p_1$.

В том случае если $p_1$ означает открыть огонь зенитными ракетами, то на естественном языке введенное корреляционное правило читается следующим образом: если вертолеты огневой поддержки «синих» вошли в зону действия зенитных ракет «зеленых», то зенитному взводу необходимо дать команду на открытие огня. Корреляционные правила вводятся в модель на этапе обучения и в совокупности образуют базу ее процедурных знаний. Помимо предписывающих, возможно использование рекомендующих (если $S^2(t)$, то лучше $p_1$, но можно $p_2$ или $p_3$) и запрещающих

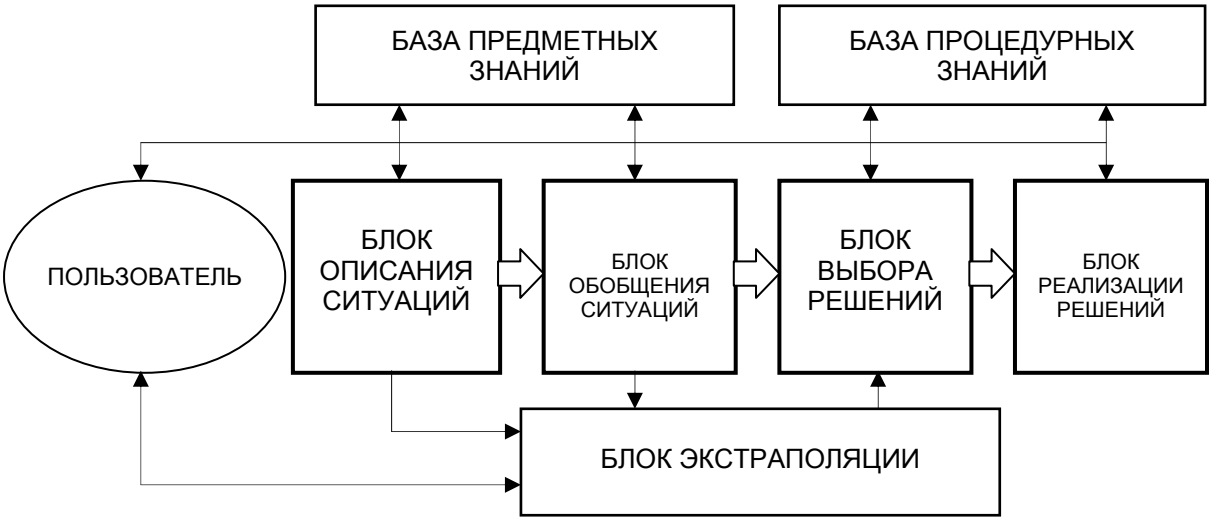


Рис. 2.23. *Функциональная схема процедуры ситуационного поиска решений*

корреляционных правил (если $S^3(t)$, то нельзя $p_1$, $p_2$,…, $p_N$), а также их комбинаций. Функциональная схема процедуры, реализующей метод ситуационного поиска решений, приведена на рис. 2.23.

Ее операционная часть включает в себя блоки описания ситуаций, обобщения ситуаций, выбора решений, экстраполяции и реализации решения. Когда обучение закончено, функционирование такой процедуры состоит в следующем. На ее вход поступает

словесное описание текущей ситуации. В блоке описания ситуаций производится формирование текстов, представляющих эту ситуацию на языке ситуационного управления. Далее в блоке обобщения ситуаций с использованием имеющихся в нем правил производится выделение ситуаций-классов.

Для выделенного класса в блоке выбора решений подбирается соответствующее корреляционное правило, на основании которого в блоке реализации решений формируются необходимые управляющие воздействия. Результат реализации вместе с принятым решением служит для дальнейшего (никогда не прекращающегося) процесса обучения. Если в процессе функционирования процедуры встречается ситуация, которую невозможно классифицировать или для выделенного класса-ситуации отсутствует корреляционное правило, то происходит обращение к блоку экстраполяции, где производится уточнение и детализация параметров ситуации, а также формирование дополнительных корреляционных правил.

### 2.2.3. Определение интегральной оценочной функции системы на основе нечетких представлений

Задача определения интегральной оценочной функции системы возникает во многих практических приложениях. Ее суть заключается в том, чтобы, зная множество частных показателей, характеризующих отдельные аспекты системы, получить обобщенную оценку ее состояния. Например, по ряду экономических, производственных, финансовых и других показателей необходимо определить интегральную эффективность субъекта рынка.

В настоящее время для решения подобных задач используются хорошо развитые методы технической диагностики, например такие как, метод Бейеса, Неймана – Пирсона, последовательного анализа, разделения в пространстве признаков, мультипликативной и аддитивной свертки, общей чертой которых является пред-

положение о независимости параметров, определяющих состояние системы, что является достаточно сильным допущением.

Ниже предлагается математический алгоритм, основанный на методе косвенной диагностики, положениях теории нечетких множеств и логико-лингвистическом подходе к моделированию систем, реализация которого позволяет определить оценочную функцию системы при условии отказа от сформулированного допущения о независимости составляющих оценочной функции [Левиатов, Захаров, 1983].

Общая идея метода заключается в имитации нестрогой (приближенной) логики мышления менеджера в процессе оценки им качества управленческих решений, в замене числовых (количественных) переменных на качественные (лингвистические), а также в использовании нечетких (эвристических) критериев и алгоритмов для установления функциональных зависимостей между входными и выходными параметрами системы. При этом базу развиваемого подхода составляют следующие аксиоматические предположения:

- менеджер имеет представление о важных переменных, описывающих динамику рынка, воспринимает взаимосвязи этих переменных и умеет оперировать правилами, связывающими переменные с управляющими решениями;
- менеджер предпочитает использовать свои собственные интуитивные правила оценки обобщенных показателей и выбора управлений, не гарантирующих математической оптимальности, но позволяющие принимать достаточно эффективные решения в сложных управленческих ситуациях;
- любая лингвистическая переменная исчерпывающим образом описывается функцией принадлежности, а логический критерий выбора состоит в том, чтобы в качестве решения выбирать такое значение переменной, в котором функция принадлежности принимает максимальное значение.

***Формулировка задачи.*** Пусть имеется некоторая абстрактная система Q и известны ее входы, выходы и внешние возбуждения (независимые внешние воздействия на систему со стороны среды, в которой происходит ее функционирование).

Пусть $X = \{x_1, x_2,\ldots, x_N\}$ – множество входных параметров системы, $Y = \{y_1, y_2,\ldots, y_K\}$ – множество выходных параметров, $Z = \{z_1, z_2,\ldots, z_M\}$ – множество внешних возбуждений, где $x_n$, $x_k$, $x_m$ – числовые переменные.

Будем считать, что если заданы параметры $x_n \in X$, $y_k \in Y$ и $z_m \in Z$, то нам известны их значения соответствующие определенному состоянию системы $s \in S$ в некоторый фиксированный момент времени t. Кроме того, для каждого параметра из множеств X, Y, Z известны его норма и допустимое отклонение от этой нормы.

Обозначим: $\delta x^*$, $\delta y^*$, $\delta z^*$ – допустимые отклонения параметров входа, выхода и внешних возбуждений от нормы; $\delta x$, $\delta y$, $\delta z$ – фактические отклонения параметров от нормы.

Моделью системы $M_Q$ будем называть кортеж

$$<\eta_x(X), \eta_y(Y), \eta_z(Z), \eta_s^{тек}(X,Y,Z)>, \qquad (2.14)$$

где $\eta_x(X)$, $\eta_y(Y)$, $\eta_z(Z)$ – оценочные функции входных, выходных и внешних параметров соответственно;

$$\eta_s^{тек}(X,Y,Z) = \eta_s(\eta_x(X), \eta_y(Y), \eta_z(Z))$$

– оценочная функция текущего состояния системы.

При выборе функций $\eta_x$, $\eta_y$, $\eta_z$ и $\eta_s$ будем исходить из того, что как сами функции, так и взаимные зависимости их аргументов нельзя задать количественно, но можно выразить качественно, используя нечеткое η-пространство со шкалами $\langle T, P, \eta\rangle$, где T – оценочная лингвистическая шкала «часто-редко», значения которой определены на интервале от «никогда» до «всегда», с числовым представлением в интервале [0, 2]; P – метрическая числовая шкала, на которой измеряются фактические значения параметров

$x_n$, $x_k$, $x_m$; $\eta$ – оценочная лингвистическая шкала, элементы которой принимают значения на интервале от «хуже не бывает» до «лучше не может быть», с числовым представлением [–1, +1].

Таким образом, как модель системы, так и оценку ее состояния будем задавать на нечетком $\eta$-пространстве в виде логико-лингвистических представлений нечетких характеристик, при конструировании которых качественным образом будем учитывать взаимные связи между параметрами системы.

Введем естественное предположение, что среди множества состояний $s \in S$ существует нормальное состояние $s^* \in S$, характеризующееся нулевыми отклонениями текущих параметров от нормальных значений. Оценочную функцию такого состояния обозначим $\eta_s^*$.

Тогда интегральная оценочная функция $\Omega$ есть кортеж

$$<\eta_s^{тес}(X,Y,Z), \eta_s^*, \rho(\eta_s^*, \eta_s^{тес}(X,Y,Z))>, \qquad (2.15)$$

где $\rho(\eta_s^*, \eta_s^{тес}(X,Y,Z))$ – функция, выражающая степень близости текущего и нормального состояний системы.

Итак, будем рассматривать цепочку $Q \rightarrow M_Q \rightarrow \Omega$, первый компонент которой есть система, второй – ее логико-лингвистическое представление в пространстве $\{T, P, \eta\}$, а третий – искомая интегральная оценочная функция. Тогда задача сводится к определению модели системы $M_Q$ через оценочные функции $\eta_x(X)$, $\eta_y(Y)$, $\eta_z(Z)$, $\eta_s^{тес}(X,Y,Z)$ и к нахождению правил вычисления $\eta_s^*$ и $\rho(\eta_s^*, \eta_s^{тес}(X,Y,Z))$.

***Логико-лингвистическое представление модели системы.*** Определим $M_Q$ через ее компоненты $\eta_x(X)$, $\eta_y(Y)$, $\eta_z(Z)$, $\eta_s^{тес}(X,Y,Z)$. В заданном пространстве оценочная функция входных параметров будет равна

$$\eta_X(X) = \varphi\,(\eta^*_x(X), \mu(x)), \qquad (2.16)$$

$$\text{где } \eta^*_x(X) = \begin{cases} \xi(1 - e^{-\upsilon(t-c_1)}); c_1 < t \le 2; \\ (-1 + e^{\upsilon(t-c_1)}); 0 < t \le c_1; \\ (1 - e^{-(t-c_2)}); c_2 < t \le 2; \\ \xi(-1 + e^{\upsilon(t-c_2)}); 0 < t \le c_2 \end{cases} \quad (2.17)$$

– максимальная и минимальная по значениям оценочной функции огибающая по шкале $\eta \in [-1,+1]$, построенная при условии, что $\delta x = 0$; $\xi \in [0,+1]$ – экспертный коэффициент разброса оценочной функции $\eta_X(X)$; $\upsilon$ – параметр уровня энтропии, характеризующий степень неопределенности знаний эксперта о системе; t – аргумент модели, совпадающий по значениям и смыслу со шкалой T; $c_1$ и $c_2$ – значения шкалы T для огибающих зависимостей по максимальным ($c_1$) и минимальным ($c_2$) значениям оценочных функций;

$$\mu(x) = 2(e^{-\lambda\frac{\delta x}{\delta x^*}} - 0{,}5), \mu(x) \in [-1,+1] \quad (2.18)$$

– функция принадлежности отношения $\delta x/\delta x^*$ к области значений лингвистической переменной «норма по параметру»;

$\lambda \in [0,+1]$ – экспертный коэффициент, характеризующий жесткость требований к допустимому отклонению параметра от его нормативного значения.

Связь между $\eta^*_x(X)$ и $\mu(x)$ выразим функцией:

$$\varphi(f_1(x), f_2(x)) = \begin{cases} f_1(x) + f_2(x), \text{sign} f_1(x) \ne \text{sign} f_2(x); \\ \min(f_1(x), f_2(x)) + F(\max(f_1(x), f_2(x))), \text{sign} f_1(x) = \text{sign} f_2(x); \end{cases} \quad (2.19)$$

где $f_1(x) \in [-1,+1]$, $f_2(x) \in [-1,+1]$;

$$F(\alpha) = \begin{cases} -2(\alpha + 1)^2; -1 \le \alpha < -0{,}5; \\ -2\alpha^2; -0{,}5 \le \alpha < 0; \\ 2\alpha^2; 0 \le \alpha < 0{,}5; \\ 2(\alpha - 1)^2; 0{,}5 \le \alpha \le 1. \end{cases}$$

Аналогично определим оценочные функции выходных параметров $\eta_y(Y)$ и внешних возбуждений $\eta_z(Z)$.

Оценочную функцию состояния системы по аналогии будем определять следующим образом:

$$\eta_s^{тес}(X,Y,Z) = \Phi\,(\varphi(\eta_x(X), \eta_y(Y), \eta_z(Z))). \qquad (2.20)$$

***Правила определения*** $\eta_s^*$ ***и*** $\rho(\eta_s^*, \eta_s^{тес}(X,Y,Z))$**.** Потребуем, чтобы $\eta_s^*$, выраженное в виде некоторой функциональной зависимости в пространстве оценочной функции, удовлетворяло следующим требованиям:

1. Зависимость должна быть непрерывно-возрастающей по значениям оценочной функции при увеличении значения аргумента $t \in T$, поскольку в противном случае нарушается метрика шкалы T.

2. Параметр уровня энтропии $\upsilon$ не должен принимать значение меньше «2» и больше «12», так как при $\upsilon < 2$ энтропия представления эксперта о системе стремится к бесконечно малой величине, а при $\upsilon = 12$ энтропия достигает своего максимального значения и дальнейшее увеличение $\upsilon$ не имеет смысла.

3. При фиксированных t между значениями оценочной функции огибающей зависимости по наибольшим ($\eta_{max}$) и наименьшим ($\eta_{min}$) значениям $\eta_x$, $\eta_y$, $\eta_z$ и $\eta_s$ должно существовать соотношение $\eta_{min} = \lambda\,(\eta_{max} + 1) - 1$,

из которого следует, что $c_2 = c_1 - \frac{2}{\lambda}\ln\lambda$.

Как показали исследования, сформулированным требованиям удовлетворяет следующее выражение:

$$\eta_s^* = \begin{cases} 1 - e^{-12\,(t-1,5)}; \text{при} \quad 1,5 \le t \le 2; \\ -1 + e^{12\,(t-1,5)}; \text{при} \quad 0 \le t < 1,5, \end{cases} \qquad (2.21)$$

которое в последующем и будем использовать для определения интегральной оценочной функции системы.

Для определения $\rho(\eta_s^*, \eta_s^{тес}(X,Y,Z))$ введем понятие меры близости состояний $s_i$, $s_j$, $(s_i, s_j) \in [-1, +1]$, установив его следующим образом:

$$\rho(s_i, s_j) = \frac{\min(\max\ \eta_{s_i}, \max\ \eta_{s_j}) - \max(\min\ \eta_{s_i}, \min\ \eta_{s_j})}{\max(\max\ \eta_{s_i}, \max\ \eta_{s_j}) - \min(\min\ \eta_{s_i}, \min\ \eta_{s_j})}. \quad (2.22)$$

Как и ранее при определении оценочной функции параметров $\eta_x(X)$, $\eta_y(Y)$, $\eta_z(Z)$, введем нечеткое трехмерное ρ-пространство в шкалах $\langle T, P, \rho \rangle$, где T, P повторяют соответствующие шкалы η-пространства, а ρ – шкала значений лингвистической переменной «мера близости состояний», определенная на интервале [+1, –1] от «строгое совпадение» (ρ = +1) до «полное несовпадение» (ρ = -1).

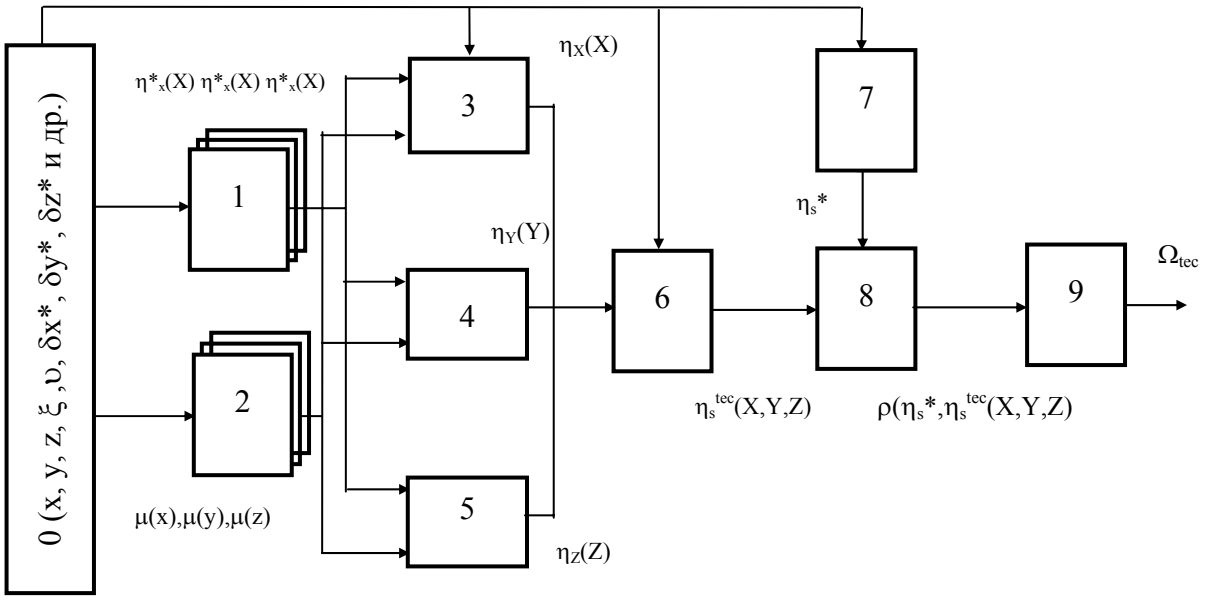


Рис. 2.24. *Алгоритм расчета интегральной оценочной функции системы*

Во введенном ρ-пространстве с учетом (2.22) интегральную оценку нечеткой близости состояний можно определить с помощью интеграла Стилтьеса:

$$\Omega(\rho(s_i, s_j)) = \int_{t=0}^{t=2} \rho(s_i, s_j)\, t dt. \quad (2.23)$$

Тогда для определения интегральной оценочной функции системы необходимо в (2.23) вместо $s_i$ подставить $\eta_s^*$, а вместо $s_j$ – $\eta_s^{tec}(X,Y,Z)$. Окончательно имеем

$$\Omega(\rho(\eta_s{*}, \eta^{tec}(X, Y, Z))) = \frac{1}{4}[2 + \int_{t=0}^{t=2} \rho(\eta_s{*}, \eta^{tec}(X, Y, Z)tdt]. \quad (2.24)$$

***Алгоритм расчета интегральной оценочной функции системы.*** Изложенные теоретические аспекты были взяты в основу построения алгоритма (рис. 2.24), программная реализация которого обеспечивает расчет интегральной оценочной функции системы. На схеме цифрами обозначены: 0 – блок ввода исходных данных; 1 – блоки расчета $\eta^*_x(X)$ $\eta^*_x(X)$ $\eta^*_x(X)$; 2 – блоки расчета $\mu(x),\mu(y),\mu(z)$; 3-5 – блоки расчета $\eta_Z(Z)$, $\eta_Y(Y)$, $\eta_X(X)$; 6 – блок расчета $\eta_s^{tec}(X,Y,Z)$; 7 – блок расчета $\eta_s{*}$; 8 – блок расчета $\rho(\eta_s{*},\eta_s^{tec}(X,Y,Z)$; 9 – блок расчета $\Omega_{tec}$.

Реализация этого алгоритма позволяет получить, например, интегральную оценку состояния рынка в зависимости от его входных и выходных параметров, также внешних возмущений, по градациям: «дестабилизированный», «крайне неустойчивый», «неустойчивый», «устойчивый», «стабильно устойчивый». Его основные преимущества (по сравнению с существующими аналогами) заключаются в следующем. Во-первых, он позволяет существенно снизить требования к квалификации экспертов, привлекаемых для исследования, поскольку от них требуются не окончательные, а лишь промежуточные оценки его состояния, например, в виде данных, необходимых для построения функций принадлежности. Во-вторых, в нем учитываются взаимозависимости между частными показателями и оценками, что важно при проведении практических исследований. В-третьих, предлагаемый алгоритм за счет изменения констант, входящих в оценочные функции, легко настраивается на конкретные условия. Более того, выходом данного алгоритма могут быть и другие градации интегральной оценки. В частности, его можно настроить на оценки типа биржевых индексов (Dow Jones, S&P, NASDAQ и др.), которые, хотя и выражаются числами, но по существу представляют собой лингвистические категории.

*  *  *

Уже рассмотренных примеров вполне достаточно для того, чтобы придти к убеждению, что логико-лингвистические методы существенно расширяют возможность формализации задач оценки и поиска решений в самых различных проблемных областях.

Их применение наиболее эффективно там, где оперируют не столько точными числами, сколько понятиями и расплывчатыми, нестрогими категориями. Конечно, отказ от числа не самый лучший вариант решения проблем, но реалии свидетельствуют о том, что таковые области объективно существуют, и так мыслят многие вполне преуспевающие управленцы. Известны имена выдающихся конструкторов и инженеров, которые при принятии стратегических решений не пользовались количественными оценками. Так, например, известному авиаконструктору А.Н. Туполеву для оценки летных качеств спроектированного самолета достаточно было посмотреть на его внешний вид. И если он говорил, что такой самолет будет летать плохо, то потом его оценка, как правило, подтверждалась результатами аэродинамических или летных испытаний.

Модель объекта и процедура поиска управленческих решений связаны одним языком, то есть от того, как будет описан объект, напрямую зависит то, как будет осуществляться оценка и поиск управленческих решений. Отсюда следует вывод: если мы способны описать изучаемый объект традиционными математическими методами, а затем сформулировать и решить задачу математической оптимизации, то прибегать к использованию логико-лингвистических методов оценки и поиска решений не следует. Как говорится, «*не надо надувать щеки и выдавать карася за порося*».

Логико-лингвистические модели расширяют сферу приложения системного анализа за пределы применимости классических методов и моделей теории управления. Оперируя понятийными категориями и нечеткими критериями, они, тем не менее, позво-

ляют моделировать достаточно сложные объекты, проводить их анализ и получать оценки, позволяющие принимать осмысленные управленческие решения, воплощая следующую точку зрения Л. Заде: *«Я считаю, что излишнее стремление к точности стало оказывать действие, сводящее на нет теорию управления и теорию систем, так как оно приводит к тому, что исследования в этой области сосредоточиваются на тех и только тех проблемах, которые поддаются точному решению. В результате многие классы важных проблем, в которых данные, цели и ограничения являются слишком сложными или плохо определенными для того, чтобы допустить точный математический анализ, оставались и остаются в стороне по той причине, что они не поддаются математической трактовке. Для того чтобы сказать что-либо существенное для проблем подобного рода, мы должны отказаться от наших требований точности и допустить результаты, которые являются несколько размытыми или неопределенными».*

Вместе с тем, нелишним будет напомнить, что, обладая достаточно широкими возможностями по описанию системных объектов, модели этого класса имеют вполне определенные границы своей применимости, обусловленные качественным характером получаемых описаний и оценок. Поэтому с практической точки зрения наилучшим следует признать комплексный подход к моделированию систем, при котором логико-лингвистические модели и мягкие вычислительные процедуры используются как инструмент структурирования проблемы и выявления ее наиболее существенных аспектов. Тем самым обеспечивается корректный переход к созданию имитационных математических моделей и применению количественных математических методов оптимизации, позволяющих не только осмыслить, но строго доказать и более убедительно подтвердить рациональность того или иного выбора.

Основные трудности создания логико-лингвистических моделей связаны с разработкой подробных и однозначных классифи-

каторов понятий изучаемой проблемной области, формулированием базовых аксиом и правил вывода, составлением алгоритмов ведения эффективного диалога с пользователем, а также с необходимостью владения современными весьма непростыми технологиями и языками программирования. Естественно, что отмеченные обстоятельства несколько сдерживают внедрение таких моделей в практику системных исследований, однако не являются непреодолимым заслоном на этом пути.

Укажем несколько перспективных направлений системных исследований, где, на наш взгляд, логико-лингвистические методы могут найти свое практическое применение.

*Создание интеллектуальных экспертных систем.* Использование логико-лингвистических методов в этом направлении открывает возможность создания диалоговых человеко-машинных экспертных систем, в которых фиксируются не окончательные заключения экспертов по тому или иному проблемному вопросу, а записываются правила, которыми они руководствуются при решении поставленных задач. Записанные на соответствующем языке представления знаний, эти правила играют роль корреляционных грамматик, позволяющих вскрывать противоречия во мнениях экспертов, анализировать побудительные мотивы принятия ими тех или иных решений, формулировать новые обобщенные оценки и рекомендации, которые в первоначальных ответах экспертов не содержались. При таком подходе повышается достоверность результатов, получаемых с помощью экспертных систем, и появляются более широкие возможности обоснования выдаваемых рекомендаций. Другой важной особенностью таких систем является возможность многократного обращения к мнениям экспертов, логика рассуждений которых записана в компьютерных программах. При этом присутствие самих экспертов необязательно.

*Управление крупномасштабными имитационными моделями операций*. Как известно, такие модели структурно включают в се-

бя большое количество взаимосвязанных программных блоков и модулей, имитирующих те или иные стороны моделируемого процесса. Порядок работы и собственно функционирование этих блоков и модулей определяется как исходными замыслами (планами) противоборствующих сторон, так и результатами моделирования на каждом из этапов. В укрупненном виде динамика развития операции может быть формально представлена в виде дискретной ситуационной сети, вершинами которой выступают макросостояния противоборствующих систем (например, выполнение ближайшей задачи, форсирование крупной водной преграды, завершение разгрома группировки противника), а дугами – направления перехода из состояния в состояние, определяемые как замыслами командований сторон, так и оценками результатов боевых действий.

Такая дискретная ситуационная сеть может рассматриваться как управляющая макромодель по отношению к имитационной модели операции, определяющая последовательность активизации ее блоков и модулей, вводимые массивы исходных данных и задающая пространственную и временную метрику моделирования. В свою очередь, результаты работы блоков и модулей определяют состояния вершин дискретной ситуационной сети и порядок изменения этих состояний. При таком подходе обеспечивается возможность непосредственного учета замыслов сторон, формирования выходных результатов моделирования в категориях, понятных операторам, а также оптимизация алгоритма работа модели в целом.

*Разработка быстродействующего математического обеспечения.* В этом направлении логико-лингвистические методы позволяют создать адаптивные по структуре алгоритмы специального математического обеспечения, позволяющие своевременно удовлетворять достаточно широкие информационные и расчетные потребности различных органов управления. Суть этих алгоритмов состоит в том, что них реализуется поиск ответа по схеме

«описание проблемной ситуации – классификация – формирование типового ответа». Иными словами, при получении запроса вместо традиционных, всегда достаточно продолжительного по времени, расчета на имитационных моделях или поиска информации в базах данных, производится быстрый выбор ответа по принципу ситуационного управления. При этом имитационные модели служат в качестве учителя, позволяя, естественно при активном участии пользователя, заблаговременно вводить в ситуационные алгоритмы классификаторы ситуации, необходимые оценки, правила принятия решения и другую информацию.

Как показывают исследования, использование таких алгоритмов позволяет на порядок повысить быстродействие систем математического обеспечения органов управления, приблизив скорость их работы к реальному масштабу времени, что важно в таких проблемных областях как управление полетами авиации, движением железнодорожного транспорта, руководство войсковыми и биржевыми операциями и другими процессами, критичными ко времени.

*Формализация процесса генерации альтернатив.* При проектировании сложных систем типовой алгоритм действий состоит в реализации следующей итеративной схемы: «формирование или генерации альтернативных вариантов → их оценка или анализ → выбор рациональных вариантов → уточнение исходных характеристик». В настоящее время в теоретическом плане наиболее полно развиты методы оценки (анализа) и выбора предпочтительных вариантов характеристик систем. При решении этих задач используется широкий арсенал методов теории исследования операций, имитационного математического моделирования, принятия решений и др. Вместе с тем вопросы формирования исходных вариантов характеристик исследуемых систем решаются, как правило, эвристически, на основе здравого смысла и опыта проектировщиков.

Такое положение, если речь идет об исследовании действительно сложной системы, зачастую приводит к тому, что из поля зрения исследователя «выпадают» рациональные варианты проектируемой системы. Это обусловлено, прежде всего, тем, что эвристическая генерация исходных альтернативных вариантов не гарантирует их полноты. Вследствие этого последующей оценке (анализу) подвергается достаточно узкий набор вариантов, среди которых нет *наиболее рационального*. В сложившейся схеме исследований это приводит к увеличению числа итераций и, следовательно, к задержке сроков выполнения проекта. С другой стороны, если сгенерировать слишком большое число исходных вариантов, то время, потребное для их оценки и анализа, может превысить все разумные пределы.

Таким образом, при проектировании сложных систем возникает проблема, суть которой в обобщенном виде сводится к нахождению компромисса между полнотой генерации исходных вариантов и ограниченной «пропускной способностью» моделей анализа (оценки) этих вариантов. Иными словами, на начальных этапах проектирования необходимо обосновать по возможности узко ограниченный набор вариантов системы, гарантирующей попадание в область глобального экстремума эффективности, и в то же время обеспечивающей их анализ (оценку) в приемлемые для практики сроки.

Поскольку речь идет о начальных этапах проектирования, для которого характерны высокая степень неопределенности исходной информации, в основном понятийный характер описаний условий применения будущей системы и недостаточно четкая определенность целей ее функционирования, то решение проблемы генерации целесообразно основывать на логико-лингвистических методах как адекватных по уровню точности располагаемым исходным данным. При этом полнота генерации исходных вариантов проектируемой системы обеспечивается за счет использования соответствующей системы классификаторов, а сужение числа

вариантов достигается путем отбрасывания заведомо неэффективных, на основе использования логических критериев.

*Формализованный анализ и проверка на непротиворечивость руководящих документов.* К таким документам относятся различного рода наставления, руководства, инструкции, положения и т.п., фактически представляющие собой словесно-описательные модели какого-либо управляемого процесса, а также модели поведения управляющих субъектов. В них содержится множество утверждений, правил, рекомендаций и других положений, которые могут быть формализованы с помощью одного из рассмотренных нами языков представления знаний. Это открывает возможность проведения формального анализа документов с цель проверки содержащихся в них положений на логическую непротиворечивость. После вскрытия противоречий производится их устранение, и цикл повторяется до тех пор, пока компьютер не выдает сообщение об отсутствии противоречий в данном документе или совокупности документов. При этом за счет использования классификаторов исключается неоднозначность понимания информации.

# ГЛАВА 3. МОДЕЛИ ОПТИМИЗАЦИИ

## 3.1. Краткий обзор моделей оптимизации: математический аспект

Пусть имеется некоторая система, функционирование которой зависит от значений вектора управляемых параметров $\overline{X} = (x_1, x_2,\ldots, x_m)$, а ее эффективность оценивается вектором численных показателей $\overline{q} = (q_1, q_2,\ldots, q_S)$, причем $q_i = q_i(\overline{X}, \overline{\omega})$, $i = \overline{1,S}$ ($\overline{\omega}$ – вектор неуправляемых параметров). Тогда модель оптимизации можно формализовать следующим образом:

$$\overline{q} = (q_1, q_2,\ldots, q_S) \xrightarrow[\overline{X\in D}]{} \text{opt}$$
$$D: q_i = q_i(\overline{X}, \overline{\omega}),\ i = \overline{1,S}, \qquad (3.1)$$
$$f_\rho(\overline{x}, \overline{\omega}) \le 0,\ \rho = 1, 2, 3,\ldots$$

Здесь символом D обозначена область, определяющая выбор допустимых решений $\overline{X}$. Обычно она и задается совокупностью ограничений типа равенства, неравенства, дискретности, функциональной связи. Выражение $\overline{q} = (q_1, q_2,\ldots, q_S) \xrightarrow[\overline{X\in D}]{} \text{opt}$, где opt = max или min есть критерий оптимизации. Выбор max или min зависит от существа решаемой задачи смысла показателей ($i = \overline{1,S}$). Неравенства $f_\rho(\overline{x}, \overline{\omega}) \le 0$ определяют функциональную связанность управляемых параметров, и указывают на то, что значения величин $(x_1, x_2,\ldots, x_m)$ нельзя выбирать произвольным образом.

В такой постановке (3.1) называется моделью оптимизации с векторным показателем в условиях неопределенности.

Неопределенность характеризуется вектором неуправляемых параметров $\overline{\omega}$. Если $\overline{\omega}$ = const, то модель (3.1) называется детерминированной и имеет вид:

$$\overline{q} = (q_1, q_2,\ldots, q_S) \xrightarrow[\overline{X\in D}]{} \text{opt}$$

$$D: q_i = q_i(\bar{x}),\ i = \overline{1,S}, \qquad (3.2)$$
$$f_\rho(\bar{x}) \le 0,\ \rho = 1, 2, 3, \ldots$$

Посмотрим, как можно свести (3.1) к виду (3.2). Если $\bar{\omega}$ представляется в виде случайных величин (или случайных функций), распределение которых приближенно известно, то в (3.1) они заменяются их математическими ожиданиями. Этот весьма грубый прием применяется при ориентировочных расчетах, когда дисперсии случайных величин $\omega$ малы, а также когда $q_i$ зависит от $\bar{\omega}$ линейно.

Наиболее общий прием сведения (3.1) к детерминированному случаю заключается в следующем. Предположим, что можно зафиксировать значения вектора $\bar{\omega}$, то есть возьмем $\bar{\omega}$ = const и решим задачу оптимизации. Полученные решения, оптимальные для заданной совокупности значений $\bar{\omega}$, назовем локально-оптимальными. Тогда путем последовательной фиксации значений вектора $\bar{\omega}$ можно получить совокупность таких локально-оптимальных решений, из которой выбирается окончательное решение на основе некоторого компромисса.

С математической точки зрения различают следующие модели оптимизации [Сысоев, 1991].

***Модель линейного программирования***. Если функции $q(\bar{x})$ и $f_\rho(\bar{x})$, характеризующие ограничения в D, линейные относительно переменных $x_j$, то (3.1) называется моделью линейного программирования и решение может быть получено симплекс-методом [Данциг, 1966].

***Модель динамического программирования.*** Процесс оптимизации заключается в поэтапном синтезе и последовательном анализе вариантов на каждом из N этапов. Показатель представляется в виде аддитивной $q(\bar{x}) = \sum_{i=1}^{N-1} q_i(\bar{x}_i, \bar{x}_{i+1})$, или мультипликативной $q(\bar{x}) = \prod_{i=1}^{N-} q_i(\bar{x}_i, \bar{x}_{i+1})$ функции векторов $\bar{x}_0$, $\bar{x}_1, \ldots, \bar{x}_N$ ($\bar{x}_N \in D$, D =

$D_0 \times D_1 \times \ldots \times D_N$).

Для решения подобных задач можно использовать схемы динамического программирования [Беллман, 1969]. В частности, эти схемы применяются и для решения задач сепарабельного программирования, когда показатели и ограничения представляются в виде суммы или произведения m функций одной переменной

$$q(\bar{x}) = \sum_{j=1}^{m} \varphi_j(x_j), \; (q(\bar{x}) = \prod_{j=1}^{m} \varphi_j(x_j)) \text{ и } f_\rho(\bar{x}) =$$

$$= \sum_{j=1}^{m} \psi_{\rho j}(x_j), \; (f_\rho(\bar{x}) = \prod_{j=1}^{m} \psi_{\rho j}(x_j)).$$

***Модель квадратичного программирования.*** В этом случае модель оптимизации характеризуется показателями в виде квадратичной функции $q(\bar{x}) = \sum_{j=1}^{m} c_j x_j + \sum_{k=1}^{m}\sum_{j=1}^{m} a_{kj} x_k x_j$, $c_j$, $a_{kj}$ – const, $x_j > 0$ и ограничениями в виде линейных функций.

***Модель выпуклого программирования.*** Если функции $q(\bar{x})$ и область допустимых решений D относится к классу выпуклых, то оптимизация может проводиться методами выпуклого программирования. Заметим, что модели линейного и квадратичного программирования являются частными случаями модели выпуклого программирования.

***Модель геометрического программирования.*** В этой модели показатели и ограничения представляются в виде полиномов вида

$$q(\bar{x}) = \sum_{k=1}^{m} c_k x_1^{a_{1k}} x_2^{a_{2k}} \ldots x_m^{a_{mk}}, \; c_k, x_k > 0.$$

***Модель дискретного программирования.*** Любая модель оптимизации, в которой имеются ограничения типа дискретности по всем координатам вектора $\bar{x}$, или по его отдельным координатам относится к классу моделей дискретного программирования. Частным случаем являются модели целочисленного программирования, в которых требуется, чтобы все или часть $x_j$ были целыми числами.

Экстремум (максимум или минимум) функции $q(\bar{x})$ может быть локальным, глобальным, внутренним, граничным, условным и безусловным. В точке $\bar{x}^*$ достигается локальный экстремум если $q(\bar{x}^*)$ есть наименьшее (наибольшее) значение функции $q(\bar{x})$ в некоторой ε-окрестности этой точки. Наименьший (наибольший) из всех локальных экстремумов называется глобальным. Понятно, что если во всей области D имеется один локальный экстремум, то он будет глобальным. В этом случае модель оптимизации называется унимодальной (одноэкстремальной) Если $q(\bar{x})$ в области D имеет несколько локальных экстремумов, то модель называется полимодальной (многоэкстремальной). Экстремумы называются граничными, если они достигаются в граничных точках области D, и внутренними, если они достигаются во внутренних точках области D. Если экстремумы $q(\bar{x})$ ищутся при отсутствии ограничений на вектор $\bar{x}$, то такие оптимизационные модели называются безусловными, в противном случае – условными.

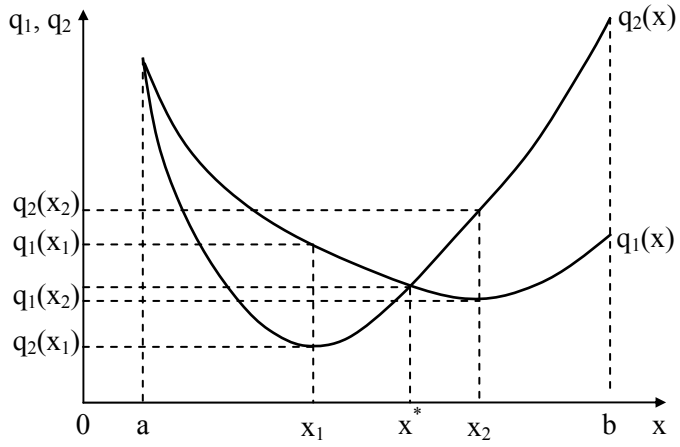

Рис. 3.1. *Поведение $q_1(x)$ и $q_2(x)$, $x \in [a, b]$*

Допустим, что в модели (3.1) используются два показателя $q_1(x)$ и $q_2(x)$ значения которых необходимо минимизировать. Область допустимых решений формально зададим отрезком оси действительных чисел $x \in [a, b]$. Характер изменения значений $q_1(x)$ и $q_2(x)$ в нормированном масштабе приведен на рис. 3.1, который показывает, что оптимумы (минимумы) по каждому из показателей достигаются в точках $x_1$, и $x_2$ соответственно для $q_1(x)$ и $q_2(x)$. Для $x \in [a, x_1]$ и $x \in [x_2, b]$ показатели $q_1(x)$ и $q_2(x)$ ведут себя согласованно, одновременно уменьшаясь либо увеличиваясь.

Рассматривая область допустимых решений $x \in [x_1, x_2]$ видим, что уменьшение значений показателя $q_1(x)$ ведет к увеличению значений показателя $q_2(x)$, то есть показатели как бы конфликтуют. Этот «конфликт» отображается в пространстве показателей $\{q\}$ в определенную область (рис. 3.2), которая называется множеством Парето.

Решения $\bar{x} \in D$, которые определяют множество Парето, будем называть не худшими. Множество не худших решений обозначим через $M_0$. В общем случае для векторной модели оптимизации введем правило, позволяющее оценивать решения, – безусловный критерий предпочтения (БКП). Решение $x_2$ безусловно лучше решения $x_1$ ($x_2 \succ x_1$) в смысле векторного показателя $\bar{q}$, если $q_i(\bar{x}_2) \le q_i(\bar{x}_1)$ для всех i, и хотя бы одно неравенство строгое. Если все $q_i(\bar{x}_2) = q_i(\bar{x}_1)$, то решение $\bar{x}_2$ эквивалентно решению $\bar{x}_1$ ($\bar{x}_2 = \bar{x}_1$).

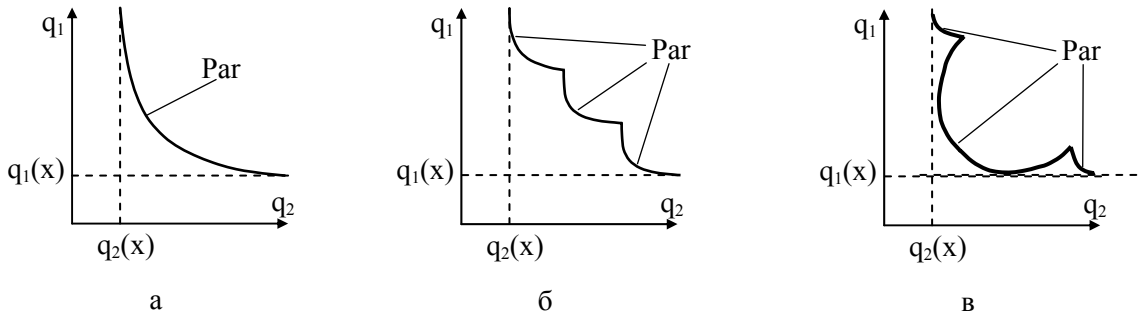


Рис. 3.2. *Примеры множества Парето (Par): а – непрерывное; б – несвязное; в – с изолированными точками*

Из всего множества D допустимых решений БКП выделяет подмножество $M_0$, не худших решений, определяющих множество Парето. Другими словами, оператор БКП реализует принцип оптимизации по Парето.

Следовательно, решением задачи оптимизации с векторным показателем формально можно считать нахождение множества $M_0$ – не худших решений. В частных случаях для этих целей используются следующие простые модели [Подиновский, Ногин, 1982]:

– для выпуклых областей D и выпуклых $q_i(x)$, $i = \overline{1,S}$

$$\text{Par} = \{\bar{q}(x^*): \sum_{i=1}^{S} \alpha_i q_i(\bar{x}^*) = \min_{\bar{x}\in D} \sum_{i=1}^{S} \alpha_i q_i(\bar{x})\};$$

– для невыпуклых областей D

$$\text{Par} = \{\overline{q}\ (x^*): \max_i \alpha_i\ q_i\ (\bar{x}^*) = \min_{\bar{x}\in D} \max_i \alpha_i q_i(\bar{x}) : q_i(\bar{x}) \geq 0\},$$

при всех $\bar{\alpha} \in A = \{(\alpha_1, \alpha_2, \ldots, \alpha_S): \sum_{i=1}^{S} \alpha_i = 1, \alpha_i > 0\}$.

Результат в виде множества $M_0$ допускает множество решений, и в этом смысле является неоднозначным. Естественно предположить, что окончательные решения следует искать среди элементов множества $M_0$.

Поэтому актуальной задачей является сужение множества $M_0$. В каждом случае такое сужение можно проводить с учетом поставленных целей исследования и конкретных условий их проведения, при помощи алгоритмов, разработка которых в настоящее время представляет важную для практики задачу. Наиболее простым способом решения указанной задачи является покрытие множества $M_0$ сетью с некоторым шагом с тем, чтобы в каждом полученном гиперпараллелепипеде оставить по одному элементу множества $M_0$. Длину шага можно выбирать из практических соображений на основе экспертного анализа. Более общим является использование алгоритмов распознавания образов и нейросетевых моделей.

Довольно часто для получения одного конкретного решения из множества Парето пользуются сведением задачи векторной оптимизации к задаче скалярной оптимизации путем выделения одного показателя (главного) и переводом остальных в разряд ограничений, либо построения глобального показателя в виде свертки показателей частных.

Таким образом, решение задачи оптимального моделирования с векторным показателем осуществляется в два этапа – это выделение области компромиссов (решений оптимальных по Парето), а затем ее сужение на основе некоторой схемы компромиссов в

частном случае до единственного решения, оптимального с точки зрения лица, принимающего решение.

## 3.2. Типовые оптимизационные модели: прикладной аспект

***Модель распределения ресурсов.*** Имеются некоторые ресурсы $R_1$, $R_2$,…, $R_m$ (сырье, рабочая сила, оборудование и др.) в количествах соответственно $r_1$, $r_2$,…, $r_m$ единиц. Эти ресурсы используются для производства продуктов $П_1$, $П_2$,…, $П_n$. Для производства одной единицы продукта $П_j$ требуется $\alpha_{ij}$ единиц ресурса $R_i$. Причем каждая единица ресурса $R_i$ стоит $C_i$ рублей, а каждая единица продукта имеет цену $Ц_j$. Спрашивается: какое количество единиц, и какой продукции необходимо производить, чтобы максимизировать некоторую меру эффективности (например, прибыль)? Методы решения подобных задач рассматриваются в [Дорожкин, Гасилов, Баркалов, 2003].

***Модель загрузки оборудования.*** Допустим, что некоторый производственный участок выпускает n видов продукции, каждый из которых может быть произведен на m типах универсального оборудования. Известна производительность $h_{ij}$ каждого i-го типа оборудования по каждому j-му виду продукции, а также лимит $t_i$ рабочего времени i-го оборудования. Требуется распределить различные виды продукции (задать некоторые пропорции) по типам оборудования таким образом, чтобы, например, обеспечивалась максимальная суммарная загрузка оборудования.

***Модель назначения.*** На некотором производственном участке, выпускающем n видов продукции, имеется также n типов универсального оборудования. При этом на каждом оборудовании может производиться любой вид продукции. Рассмотрим матрицу $\left\|C_{ij}\right\|$ $(I = \overline{1,n}\,;\ j = \overline{1,n}\,)$, где $C_{ij}$ – параметры, определяющие эффективность использования i-го типа оборудования для производства j-го вида продукции (производительность, стоимость, надежность

и др.). Тогда задача заключается в таком выборе n элементов из матрицы $\|C_{ij}\|$ по одному из каждой строки и каждого столбца, чтобы максимизировать общую меру эффективности оборудования (в частности, их суммарную производительность). Другими словами, требуется оптимальным образом назначить типы оборудования для производства каждого вида продукции.

***Модель упорядочения.*** Пусть имеется n видов продукции, которые требуется обработать на m типах оборудования. Каждый вид продукции имеет свой технологический маршрут обработки. Известно время $t_{ij}$ ($i = \overline{1,n}$; $j = \overline{1,m}$), необходимое для обработки i-го вида продукции на j-ом типе оборудования. Причем считается, что одновременная обработка нескольких видов продукции невозможна. При таких условиях у некоторого оборудования может образоваться очередь ждущих обработки видов продукции. Требуется определить такую очередность (последовательность) обработки продукции на каждом типе оборудования, которая: обеспечивает минимум суммарной продолжительности обработки всех видов продукции или минимум общего запаздывания обработки, или минимум запаздывания, или минимум потерь, обусловленных запаздыванием, и др. Запаздывание определяется как разность между фактическим и директивным сроком обработки по каждому виду продукции.

***Модель управления запасами.*** С увеличением запасов в производстве, с одной стороны, вырастают затраты на их хранение, а с другой – снижаются потери из-за возможной их нехватки. Требуется определить такой уровень запасов, который обеспечивает максимальную эффективность производства (например, минимум ожидаемых затрат по хранению запасов, а также потерь из-за их дефицита; максимум производительности системы и др.).

Рассмотрим одну из простейших постановок такой задачи. Пусть в моменты времени 1, 2, …, n, … поступают заявки на некоторый товар. Размеры этих заявок носят случайный характер:

спрос в момент k есть случайная величина $\xi_k$. Будем считать, что величины $\xi_k$ при разных k независимы и одинаково распределены с плотностью p(x), которую мы считаем положительной при $x > 0$. Для удовлетворения спроса имеется возможность запастись некоторым количеством товара, a, кроме того, в каждый момент времени n можно сделать дополнительный заказ u(k), который поступит в момент n + 1. Товар, заказанный ранее, который доставят через единицу времени после заказа, обходится в h денежных единиц за единицу товара. Если в какой-то момент спрос превышает предложение, то имеется возможность получить товар тотчас же и тем самым удовлетворить спрос. Однако в этом случае товар поступит по цене H за единицу. Естественно, что $H > h$. Задача состоит в выборе такой стратегии предварительных заказов $u_0, u_1,\ldots, u_{N-1}$ в течении N единиц времени, чтобы удовлетворить спрос с наименьшими общими затратами.

Обозначим $x_n$ количество товаров, оставшихся после удовлетворения n-й заявки $\xi_n$. Тогда $x_{n+1} = x_n + u_n$, если только правая часть этого равенства положительна. Если спрос $\xi_{n+1}$ в момент n + 1 превышает количество товара $x_n + u_n$, имеющегося к моменту n + 1, то недостаток покрывается за счет немедленной закупки по штрафной цене, и в этом случае $x_{n+1} = 0$. Если обозначить $f^+(x)$ функцию, равную f(x) при $f(x) > 0$ и нулю при $f(x) \le 0$, то связь $x_{n+1}$ и $x_n$ можно записать так:

$$x_{n+1} = (x_n + u_n - \xi_{n+1}). \qquad (3.3)$$

Тогда при фиксированной последовательности $u = (u_0, u_1, \ldots, u_{N-1})$ предварительных закупок наши затраты составят следующую сумму:

$$h\sum_{k=0}^{N-1} u_k + H\sum_{k=0}^{N-1}(\xi_{k+1} - x_k - u_k)^+ = L_N[u], \qquad (3.4)$$

где первое слагаемое есть плата за предварительно заказанные товары, а второе – стоимость штрафных закупок.

Поскольку вся сумма $L_N[u]$ – случайная величина, то в качестве характеристики последовательности u примем среднее значение этой случайной величины $ML_N[u]$. Тогда задача будет состоять в выборе такого допустимого управления $u = (u_0, u_1,\ldots, u_{N-1})$, чтобы минимизировать среднюю стоимость товаров за время [0, N]. Поясним, какие управления считаются допустимыми. Прежде всего, все компоненты вектора u – неотрицательные величины. Далее, величина заказа в момент k вычисляется по ходу процесса до момента k включительно, то есть по $x_1, x_2,.. , x_k$. Это означает, что из всех возможных управлений могут быть выбраны только те, которые зависят от текущего состояния процесса, то есть оптимальная величина заказа в k-й момент – $u_k$ может быть найдена среди функций, зависящих от k и $x_k$:

$$u_k^* = v(k, x_k)\big|_{h,H,p(x),N} . \qquad (3.5)$$

Если используются управления вида (3.5), то последовательность $x_n$, определяемая равенством (3.3), является марковским процессом. В этом случае существует простая оптимальная стратегия предварительных заказов: на каждом этапе существует свой критический уровень запасов $a_k$, который тем ниже, чем ближе к заключительному этапу, и наша политика заказов должна сводиться к поддержанию этого уровня. Величины $a_k$ рассчитываются с помощью стандартных формул теории марковских процессов [Дынкин, Юшкевич, 1967].

Никаких принципиальных трудностей не возникает, если в условиях предыдущей задачи будет фигурировать не один, а несколько видов товара. В этом случае оптимальная политика также состоит в поддержании критического уровня запасов каждого из этих видов. Аналогично рассматривается задача о создании запасов, когда заказанные товары поступают с запаздыванием на фиксированное число этапов. Новые эффекты возникают, если время запаздывания – случайная величина.

В ряде практических задач более естественна другая структу-

ра функции штрафов. Например, нежелательно, чтобы запас превосходил некоторую величину, а дефицит товаров не превышал некоторую другую константу. Тогда оптимальная политика создания запасов должна состоять в том, чтобы случайный процесс $x_n$ – количество имеющихся или недостающих товаров в момент n не выходил за эти границы по возможности с большей вероятностью. Или, другая постановка, – чтобы среднее время до выхода за эти границы было бы минимальным.

***Транспортная модель.*** В пунктах $A_1, A_2, \ldots, A_m$ производится некоторый однородный продукт в количествах соответственно $a_1, a_2, \ldots, a_m$ единиц. Этот продукт необходимо доставить в пункты назначения $B_1, B_2,\ldots, B_n$, потребляющих этот продукт, соответственно в количествах $b_1, b_2,\ldots, b_n$ единиц. При этом, как правило, считается, что общий запас груза в пунктах отправления равен суммарной потребности в пунктах потребления $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$. В этих условиях требуется найти такой план перевозок, который доставляет в некотором смысле оптимум вектору показателей эффективности. В частности, если известны стоимости перевозки единицы продукции из $A_i$ в $B_j$, то можно искать такой план, который минимизирует общую стоимость перевозок.

***Модель оптимального проектирования.*** В процессе проектирования, например, некоторой технологической системы, требуется определить ее параметры $\overline{x} = (x_1, x_2,\ldots, x_m)$ таким образом, чтобы максимизировать эффективности функционирования системы, такие как надежность, быстродействие, точность, производительность и др., минимизировать приведенные затраты, занимаемую площадь, время производственного цикла и др. Решение задачи обеспечивается варьированием параметров x в некоторой допустимой области D, которая формируется системой ограничений типа равенств, неравенств, дискретности, и определяется требованиями технологии.

***Модель обнаружения разладки***. Предположим, что некая автоматическая линия выпускает одну деталь в единицу времени, и эта деталь характеризуется одним параметром, скажем, своим диаметром. Если линия хорошо отлажена, то этот диаметр в среднем равен некоторой постоянной величине, но за счет случайных ошибок возможны отклонения от среднего, так что диаметр детали естественно считать случайной величиной, имеющей некоторую плотность $p_0(x)$. Для разных деталей эти величины независимы.

В случайный момент времени $\theta$ происходит разладка линии, что приводит к тому, что диаметры деталей, выпущенных в момент $\theta$ и позже, уже имеют распределение, отличное от $p_0(x)$. Пусть $p_1(x)$ – плотность нового распределения. Таким образом, диаметр n-й детали $\xi_n$ имеет плотность $p_0(x)$ при $n < \theta$ и $p_1(x)$ при $n \geq \theta$.

Задача состоит в том, чтобы по наблюдениям за размерами деталей как можно скорее обнаружить разладку линии, после чего ее остановить и переналадить. Отклонения размеров деталей от стандарта могут объясняться двумя причинами. Во-первых, случайными ошибками, поэтому объявление о разладке может быть ложной тревогой – разладки линии к этому моменту может и не быть. Во-вторых, разладка линии может быть следствием поломок оборудования, либо их износом. Тогда, если быть очень осторожным, можно пропустить момент разладки, приняв отклонения в размерах детали за случайный сбой, и линия будет долго работать в разлаженном режиме. Плохо как то, так и другое. Поэтому целесообразно найти некую «золотую середину», основываясь на оценках стоимости переналадки линии и стоимости ущерба за время работы разлаженной лини. Математическая формулировка и варианты решения такой задачи рассматриваются в следующих работах [Дынкин, Юшкевич, 1967, Ширяев, 1969].

***Классическая модель оптимального управления***. Рассматривается некоторый объект (пусть это будет ракета), состояние которого в каждый текущий момент времени характеризуется n действительными числами ($x^1$, $x^2$,…, $x^n$). Векторное пространство X векторной переменной $x = (x^1, x^2,\ldots, x^n)$ является фазовым пространством рассматриваемого объекта. Поведение (движение) объекта заключается в том, что переменные $x^1$, $x^2$,…, $x^n$ меняются с течением времени.

Предполагается, что движением объекта можно управлять, то есть, что объект снабжен некоторыми «рулями», от положения которых зависит его движение. Положение «рулей» характеризуется точками u некоторой области управления U, которая может быть любым множеством некоторого r-мерного евклидова пространства $E_r$. Задание точки $u = (u^1, u^2,\ldots, u^r) \in U$ равносильно заданию системы числовых параметров $u^1$, $u^2$,…, $u^r$. В приложениях обычно полагают, что U является замкнутой областью пространства $E_r$. В частности, область управления может быть кубом или каким-либо другой замкнутой фигурой r-мерного пространства переменных $u^1$, $u^2$,…, $u^r$.

Физический смысл рассмотрения замкнутой и ограниченной области управления заключается в том, что управляющие параметры (количество подаваемого в двигатель топлива, температура, напряжение и т.п.) не могут принимать сколь угодно больших значений. Кроме того, в силу технической конструкции, между управляющими параметрами $u^1$, $u^2$,…, $u^r$ могут существовать связи, выражаемые одним или несколькими уравнениями вида $\varphi(u^1, u^2,\ldots, u^r) = 0$. Например, если имеются два управляющих параметра $u^1$, $u^2$, которые в силу конструкции объекта имеют вид $u^1 = \cos\varphi$, $u^2 = \sin\varphi$, где $\varphi$ – некоторый (произвольно задаваемый) угол, то областью управления будет окружность $(u^1)^2 + (u^2)^2 = 1$.

Каждая функция u = u(t), определенная на некотором временном отрезке $t_0 \le t \le t_1$ и принимающая значения в области U, на-

зывается управлением. Так как U есть множество в пространстве управляющих параметров $u^1$, $u^2$,…, $u^r$, то каждое управление $u(t) = (u^1(t), u^2(t),\ldots, u^r(t))$ есть вектор-функция (заданная на отрезке $t_0 \le t \le t_1$), значения которой лежат в области U. В зависимости от характера решаемой задачи на управления накладываются различные условия – непрерывности, выпуклости, кусочной непрерывности, дифференцируемости и другие. Управления $u(t) \in U$, удовлетворяющие этим условиям, называются допустимыми управлениями.

Предполагается, что закон движения объекта (и закон воздействия «рулей» на это движение) записывается в виде системы дифференциальных уравнений

$$\frac{dx^i}{dt} = f^i(x^1, x^2, ..., x^n, u^1, u^2, ..., u^r) = f^i(x, u) \qquad (3.6)$$

или, в векторной форме,

$$\frac{dx}{dt} = f(x, u), \qquad (3.7)$$

где f(x, u) – вектор с координатами $f^1(x, u)$, $f^2(x, u)$,…, $f^n(x, u)$.

Функции $f^i$ предполагаются непрерывными по переменным $x^1$, $x^2$,…, $x^n$, u и непрерывно дифференцируемыми по $x^1$, $x^2$,…, $x^n$.

Если задан закон управления, то есть, выбрано некоторое допустимое управление u = u(t), то уравнение (13.7) принимает вид

$$\frac{dx}{dt} = f(x, u(t)), \qquad (3.8)$$

откуда (при любых начальных условиях $x(t_0) = x_0$) однозначно определяется закон движения объекта x = x(t), то есть решение уравнения (13.8), определенное на некотором отрезке времени.

Предположим, что задана еще одна функция $f^0(x^1, x^2, \ldots, x^n, u) = f^0(x,u)$, определенная и непрерывная вместе с частными производными $\frac{\partial x}{\partial t}, i = 1,2,..., n$, на всем пространстве

$X \times U$. Физический смысл этой функции может быть самым различным, но в большинстве практических случаев ей выражаются затраты на управление, например, в виде расхода топлива, электроэнергии, финансовых средств и т. п.

Тогда классическая задача отыскания оптимальных управлений формулируется следующим образом. В фазовом пространстве X даны две точки $x_1$ и $x_2$. Необходимо среди всех допустимых управлений $u = u(t)$, переводящих фазовую точку из положения $x_0$ в положение $x_1$ (если такие управления существуют), найти такое, для которого функционал

$$J = \int_{t_0}^{t_1} f^0(x(t), u(t))dt \qquad (3.9)$$

принимает наименьшее возможное значение; здесь $x(t)$ – решение уравнения (13.7) с начальным условием $x(t_0) = x_0$, соответствующее управлению $u(t)$, а $t_1$ – момент прохождения этого решения через точку $x_1$.

Управление $u(t)$, дающее решение сформулированной задачи, называется оптимальным, а соответствующая траектория $x(t)$ – оптимальной траекторией.

Важной разновидностью сформулированной выше задачи является случай, когда $f^0(x,u) \equiv 0$, и, соответственно, функционал (3.9) принимает вид:

$$J = t_1 - t_0. \qquad (3.10)$$

В этом случае оптимальность управления $u(t)$ означает минимальность времени перехода объекта их положения $x_0$ в положение $x_1$. Такая задача называется задачей об оптимальном быстродействии. Решение таких задач основывается на принципе максимума [Понтрягин, 1976].

***Модель преследования.*** Предположим, что в n-мерном фазовом пространстве X движутся два управляемых объекта, один из которых будем называть «преследующим», а другой – «преследуемым». Движение каждого из этих объектов подчиняется своей

собственной системе дифференциальных уравнений со своим собственным управляющим параметром. Управляющий параметр, область управления и траекторию движения преследующего объекта обозначим соответственно через u, U, x(t), а для преследуемого – символами v, V, x(t).

Пусть u(t), v(t) – некоторые допустимые управления, а x(t), y(t) – соответствующие им траектории с начальными условиями

$$X(0) = x_0,\ y(0) = y_0 \qquad (3.11)$$

Если для некоторого $t_1 > 0$ выполняется равенство $x(t_1) = y(t_1)$, то число $t_1$ называется моментом встречи, а сам факт выполнения равенства $x(t_1) = y(t_1)$ – встреча. Вообще, если управления u(t) и v(t) выбраны произвольно, то встречи может не произойти ни при каком $t > 0$. Если же встреча происходит, то говорят, что u(t) является преследующим управлением (для заданного управления и заданных начальных условий $x_0$, $y_0$). При этом для заданных $x_0$, $y_0$ и v(t) и выбранного управления u(t) может произойти не одна встреча. Наименьшее положительное число $t_1$, являющееся моментом встречи, называется временем преследования, соответствующим управлениям u(t), v(t). Время преследования обозначим через $T_{u,v}$. В дальнейшем начальные условия (13.11) предполагаются фиксированными (в связи с чем в обозначение времени преследования начальные условия $x_0$, $y_0$ не входят).

Будем исходить из того, что преследующий объект обладает следующим свойством: для любого заданного управления v(t) существует (при заданных начальных условиях (3.11)) преследующее управление u(t). Тогда, если управление v(t) преследуемого объекта выбрано, то можно поставить вопрос о нахождении такого преследующего управления u(t), чтобы соответствующее время преследования $T_{u,v}$ принимало минимальное значение. Будем предполагать, что для любого допустимого управления v(t) существует допустимое управление u(t), осуществляющее мини-

мум времени преследования. Этот минимум обозначим через $T_v = \min_u T_{u,v}$.

Будем далее полагать, что существует допустимое управление v(t), осуществляющее максимум величины $T_v$. Этот максимум обозначим символом T:

$$T = \max_v T_v = \max_v (\min_u T_{u,v}). \qquad (3.12)$$

Задача заключается в том, чтобы выбрать такую пару допустимых управлений u(t), v(t), что для соответствующего времени преследования $T_{u,v}$ выполняется равенство $T_{u,v} = T$. Такие управления u(t), v(t) называются оптимальными управлениями, а соответствующие им траектории x(t), y(t) (с начальными значениями (3.11)) – оптимальными траекториями.

Итак, управление u (при любом заданном управлении v(t)) выбирается таким образом, чтобы, по возможности, ускорить встречу преследующего объекта с преследуемым, выбор же управления v подчинен задаче максимально отдалить (во времени) момент встречи. Отметим, что при выборе управления u(t), определяющего движение преследующего объекта, всякий раз предполагается заранее известным управление v(t) для преследуемого объекта. В соответствии с этим, при определении величины T сначала берется минимум по возможным управлениям u(t) при некотором фиксированном v(t), а затем берется максимум по возможным управлениям v(t).

Введем следующие допущения:

1) Движение преследующего объекта описывается в пространстве X линейным векторным уравнением

$$\frac{dx}{dt} = f(x,u) = Ax + Bu + c, \qquad (3.13)$$

для которого выполнено условие общности положения, а соответствующая область управления U представляет собой замкнутый выпуклый ограниченный многогранник в пространстве $E_r$ переменной $u = (u^1, \ldots, u^r)$.

2) Движение преследуемого объекта описывается векторным уравнением

$$\frac{dy}{dt} = g(y, v, t), \tag{3.14}$$

а соответствующая область управлений V является множеством s-мерного пространства переменной $v = (v^1,\ldots, v^s)$. Векторная функция g(y, v, t) предполагается непрерывной по переменным y, v, t и непрерывно-дифференцируемой по координатам $u^1,\ldots, u^r$ точки u.

3) В качестве класса допустимых управлений (как для u, так и для v) примем множество всех кусочно-непрерывных управлений.

Кроме того, введем в рассмотрение два вспомогательных вектора

$$\psi = (\psi_1,\ldots, \psi_n),\ \chi = (\chi_1,\ldots, \chi_n)$$

и две гамильтоновы функции

$$H_1(\psi, x, u) = \sum_{\alpha=1}^{n} \psi_\alpha f^\alpha(x, u) = (\psi, f(x, u)),$$

$$H_1(\chi, y, v) = \sum_{\alpha=1}^{n} \chi_\alpha g^\alpha(y, v, t) = (\chi, g(y, v, t)),$$

соответствующие преследующему и преследуемому объектам. С помощью функций $H_1$ и $H_2$ запишем две системы уравнений для вспомогательных неизвестных $\psi_i$ и $\chi_i$:

$$\frac{d\psi_i}{dt} = -\frac{dH_1}{dx^i}, i = 1,2,\ldots, n, \tag{3.15}$$

$$\frac{d\chi_i}{dt} = -\frac{dH_2}{dy^i}, i = 1,2,\ldots, n. \tag{3.16}$$

Тогда нижеследующая теорема [Понтрягин, 1976] дает необходимое условие оптимальности для рассматриваемой задачи.

*Теорема.* Пусть u(t), v(t) – оптимальная пара управлений, x(t), y(t) – соответствующая пара оптимальных траекторий (см. уравнения (13.13), (13.14)) и T – время преследования. Тогда сущест-

вуют такие нетривиальные решения $\psi(t)$ и $\chi(t)$ систем (3.15), (3.16), соответствующие функциям u(t), v(t), x(t), y(t), что:

1° для всех t, $0 \le t \le T$, выполняются условия максимума

$$\max_{u \in U} H_1(\psi(t), x(t), u) = H_1(\psi(t), x(t), u(t)),$$

$$\max_{v \in V} H_2(\chi(t), y(t), v) = H_2(\varphi(t), y(t), v(t));$$

2° в момент t = T выполняются условия

$$H_1(\psi(T), x(T), u(T)) \ge H_2(\chi(T), y(T), v(T)),$$

$$\psi(T) = \chi(N).$$

## 3.3. НЕЛИНЕЙНЫЕ МОДЕЛИ ОПТИМИЗАЦИИ

***Общая модель нелинейного программирования.*** Не теряя общности, эту модель будем записывать следующим образом:

$$q(\bar{x}) \xrightarrow[\bar{x} \in D]{} \min, \qquad (3.17)$$

$$D = \{\bar{x} : h_k(\bar{x}) \le 0,\ k = \overline{1,P};\ x_{i\,min} \le x_i \le x_{i\,max};\ i = \overline{1,m}\}.$$

где $q(x_1,\ldots, x_m)$, $h_k(x_1,\ldots, x_m)$, $k = \overline{1,P}$ – нелинейные функции.

Предположим, что в модели (3.17) ограничения в области D заданы в виде равенств, то есть $h_k(\bar{x}) = 0$, $k = \overline{1,P}$. Очевидно, что такая модель является частным случаем, так как каждое равенство $h_k(\bar{x}) = 0$ можно заменить двумя неравенствами $h_k(\bar{x}) \le 0$, $h_k(\bar{x}) \le 0$, а ограничения-неравенства представимы в форме равенств введением дополнительных переменных $z_k$ так, чтобы $h_k(\bar{x}) + z_k = 0$, $k = \overline{1,P}$.

*Геометрическая интерпретация*. Вектор $\bar{x}$ в m-мерном пространстве переменных определяет точку с координатами $x_1,\ldots, x_m$. Функция q в каждой точке такого пространства образует поверхности равного уровня, уравнения которых записываются в виде

$$q(\bar{x}) = \text{const}. \qquad (3.18)$$

При m = 2 уравнения (3.18) образуют так называемые линии уровня (рис. 3.3).

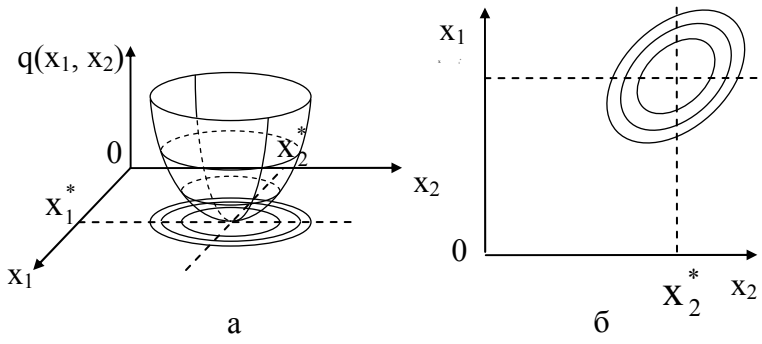


Рис. 3.3. *Образование линий уровня:*
$\bar{x}^* = (x_1^*, x_2^*)$ *– оптимальное решение;*
*а - общее; б - проекция на плоскость*

Рассмотрим теперь, как в пространстве параметров $\bar{x}$ представляются ограничения.

Ограничения типа равенств $h_k(x_1, \ldots, x_m) = 0$, $k = \overline{1,P}$ уменьшают размерность пространства параметров, так как переменные $x_1,\ldots, x_m$ могут выбираться на поверхности, определяющейся пересечением гиперповерхностей $h_k(\bar{x})$, $k = \overline{1,P}$ всех ограничений одновременно, Например, при $m = 3$ решение находится на поверхности $h_k(x_1, x_2, x_3) = 0$, которая и определяет двумерное пространство оптимизируемых параметров (рис. 3.4).

Ограничения типа неравенств $h_k(x_1,\ldots, x_m) \le 0$ образуют (высекают) в m-мерном пространстве параметров сферический многогранник, определяющий область допустимых решений D. Уравнения $h_k(\bar{x}) = 0$ являются уравнениями границы, разделяющей пространство параметров на две части – в одной из них $h_k(\bar{x}) \le 0$, в другой – $h_k(\bar{x}) \ge 0$. Сказанное иллюстрируется рис. 3.5. Здесь нанесены линии уровня для функции $q(x_1, x_2)$ и область допустимых решений

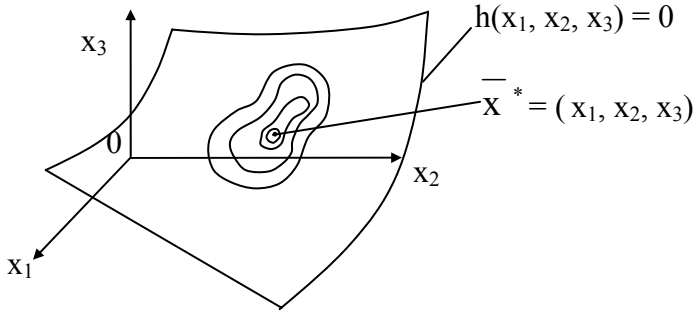


Рис. 3.4. *Пример ограничений типа равенства,*
$\bar{x}^* = (x_1, x_2, x_3)$ *– оптимальное решение*

$$D = \{x_1 > 0;\ x_2 > 0;\ h_1(x_1, x_2) \le 0;\ h_2(x_1, x_2) \le 0;\ h_3(x_1, x_2) \le 0\}.$$

При этом минимум показателя качества (критерия) достигается на границе области D. В точке $\bar{x}^*$ достигается локальный минимум, если $q(\bar{x}^*)$ наименьшее значение функции в некоторой ε окрестности этой точки. Наименьший из всех локальных минимумов будет глобальным. Понятно, что если во всей области D имеется один локальный минимум, то он будет глобальным (см. рис. 3.3-3.5).

Одной из важных геометрических характеристик многомерной функции $q(\bar{x})$ является ее градиент.

$$\text{grad } q(\bar{x}) = \left( \frac{\partial q(\bar{x})}{\partial x_1}, \frac{\partial q(\bar{x})}{\partial x_2}, ..., \frac{\partial q(\bar{x})}{\partial x_m} \right) \qquad (3.19)$$

Градиент скалярной функции $q(\bar{x})$ в некоторой точке – это вектор, направленный в сторону наибольшего увеличения функции и ортогональный поверхности (линии) уровня в этой точке. Противоположный вектор называется антиградиентом. Он направлен в сторону наискорейшего убывания функции $q(\bar{x})$ (рис. 3.6).

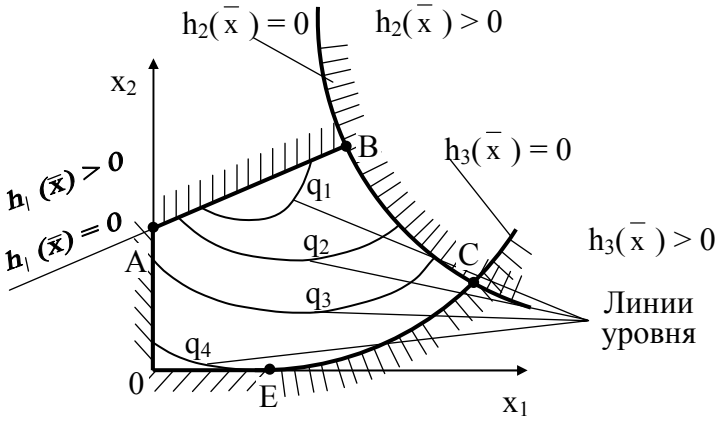


Рис. 3.5. *Пример ограничений типа неравенств*: $q_1 > q_2 > q_3 > q_4$; $\bar{x}^*$ – *оптимальное решение*; OABCE – *область допустимых решений*

Из (3.13) следует, что вектор grad $q(\bar{x})$ можно получить для широкого класса аналитически заданных функций путем вычисления соответствующих частных производных $\partial q/\partial x_i$, $i = \overline{1,m}$. Однако в практических задачах такой подход обычно невозможен, так как, с одной стороны, функции $q(\bar{x})$, как правило, недифференцируемы, с другой стороны – практические задачи решаются на компьютерах. Это требует применения разно-

стных методов для приближенной оценки частных производных. Известны два таких метода [Альянах, 1988]:

$$\left.\frac{\partial q(\bar{x})}{\partial x_i}\right|_{\bar{x}=\bar{x}_k} \approx \frac{q\left(x_1^k,\ldots,x_i^k+\Delta x_i,\ldots,x_m^k\right)}{\Delta x_i},\ \forall i=\overline{1,m}; \qquad (3.20)$$

$$\left.\frac{\partial q(\bar{x})}{\partial x_i}\right|_{\bar{x}=\bar{x}_k} \approx \frac{q\left(x_1^k,\ldots x_i^k+\Delta x_i,\ldots x_m^k\right)-q\left(x_1^k,\ldots x_i^k-\Delta x_i,\ldots x_m^k\right)}{2\Delta x_i},\ \forall i=\overline{1,m}, \qquad (3.21)$$

где $\bar{x}_k = (x_1^k, x_2^k, \ldots, x_m^k)$ – k-я точка, в которой вычисляется градиент; $\Delta x_i$ – величина шага по i-й координате в окрестности точки $\bar{x}_k$.

Точность формулы (3.21) выше, чем формулы (3.20). Однако для ее использования требуется большее число арифметических

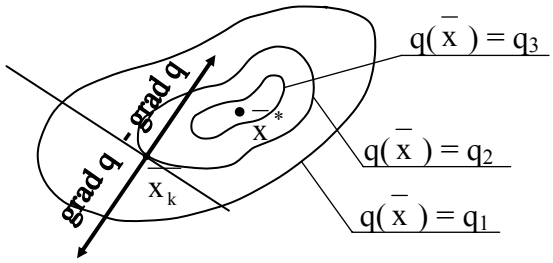


Рис. 3.6. *Геометрическая интерпретация градиента в точке* $\bar{x}_k$ *:* $q_1 > q_2 > q_3 > \ldots$

вычислений (практически в два раза), что необходимо учитывать при проведении практических расчетов.

*Численная схема отыскания экстремума*. Существуют два подхода к решению задачи (3.17).

Первый (классический) заключается в замене задачи на экстремум задачей поиска решений системы уравнений вида

$$\frac{\partial q}{\partial x_i} = 0;\ i=\overline{1,m}; \qquad (3.22)$$

при нахождении безусловного экстремума (ограничения отсутствуют) или задачей

$$\frac{\partial L}{\partial x_i} = \frac{\partial q}{\partial x_i} + \sum_{k=1}^{P}\lambda_k\frac{\partial h_k}{\partial x_i},\ i=\overline{1,m}, \qquad (3.23)$$

$$\frac{\partial L}{\partial \lambda_i} = h_k(\bar{x}) = 0,\ k = \overline{1,P}\ .$$

где $\bar{\lambda} = (\lambda_1,\ldots, \lambda_k)$ – вектор произвольных множителей, если ограничения заданы в виде равенств.

В (3.23) $L = L(\bar{x}, \bar{\lambda})$ – называется функцией Лагранжа. Оптимизация здесь сводится к решению системы (m + P) уравнений относительно неизвестных $x_1,\ldots, x_m, \lambda_1,\ldots, \lambda_P$.

Второй подход связан с так называемыми численными методами поиска. Здесь требуется построить оценку $\bar{x}^* \in D$ оптимального решения $\bar{x}^*$ (предполагается, что оно существует) на основе конечного числа k значений функции $q(\bar{x})$, последовательно вычисляемых в точках $\bar{x}_0, \bar{x}_1, ..., \bar{x}_k \in D$, причем эти значения образуют убывающую сходящуюся последовательность $q(\bar{x}_0) > q(\bar{x}_1) > ... > q(\bar{x}_k = \bar{x}^*)$. Последовательность $\{\bar{x}_k\}$ называется релаксационной, а методы ее построения – методами спуска.

Тогда, в общем случае, можно предложить итерационную формулу построения $\{\bar{x}_k\}$:

$$\bar{x}_{k+1} = \bar{x}_k + \bar{\alpha}_k \bar{P}_k,\ k = 0, 1, 2, ... \qquad (3.24)$$

где $\bar{P}_k$ – направления спуска; $\bar{\alpha}_k$ – длина шага спуска вдоль выбранного направления $\bar{P}_k$.

Степень близости оценки $\bar{x}_*$ к оптимальному решению $\bar{x}^*$ можно задать, например, в виде $|\bar{x}_* - \bar{x}^*| \le \varepsilon$ или $|q(\bar{x}^*) - q(\bar{x}_*)| \le \varepsilon$, где $\varepsilon > 0$ – заданная точность.

Одним из показателей эффективности численных методов спуска может служить их скорость сходимости. Говорят:

– о линейной сходимости (сходимость со скоростью геометрической прогрессии), если

$|\bar{x}_k - \bar{x}^*| \le \mu |\bar{x}_{k-1} - \bar{x}^*|,\ 0 < \mu < 1;$

– о сверхлинейной сходимости, если

$|\bar{x}_k - \bar{x}^*| \le \mu_k |\bar{x}_{k-1} - \bar{x}^*|$, где $\mu_k \to 0$ при $k \to \infty$;

– о квадратичной сходимости, если

$\left|\bar{x}_k - \bar{x}^*\right| \le C\left|\bar{x}_{k-1} - \bar{x}^*\right|^2,\ C \ge 0.$

На рис. 3.7 приведена схема вычислений для одной итерации такого поиска в соответствии с формулой (3.24).

***Градиентные модели.*** В этих моделях используются свойства градиента. Выбирая в формуле (3.24) в качестве направления спуска антиградиент получим следующий итерационный процесс поиска в координатной форме

$$\begin{cases} x_1^{k+1} = x_1^k - \alpha_k \left.\dfrac{\partial q}{\partial x_1}\right|_{\bar{x}=\bar{x}_k}, \\ x_2^{k+1} = x_2^k - \alpha_k \left.\dfrac{\partial q}{\partial x_2}\right|_{\bar{x}=\bar{x}_k}, \\ \ldots\ldots\ldots\ \ldots\ldots\ldots\ \ldots\ldots\ldots\ . \\ x_m^{k+1} = x_m^k - \alpha_k \left.\dfrac{\partial q}{\partial x_m}\right|_{\bar{x}=\bar{x}_k}, \end{cases} \qquad (3.25)$$

где частные производные $\partial q/\partial x_i$, $i = \overline{1,m}$ вычисляются либо по формуле (3.20), либо (3.21): $\bar{x}_k = (x_1^k, x_2^k, \ldots, x_m^k)$.

Существуют различные способы выбора шага $\alpha_k$ в формулах (3.25). При этом исходят из того, что, с одной стороны, длина шага не должна быть слишком большой, чтобы не «проскочить» экстремум (алгоритм может зациклиться), с другой – длина шага не должна быть слишком малой, чтобы итерационная процедура оптимизации не оказалась медленной. Ниже рассмотрим два возможных способа выбора шага $\alpha_k$.

Первый способ заключается в том, что фиксируется некоторое значение $\alpha_k > 0$, и процедура оптимизации реализуется для $k = 0, 1, 2, \ldots$ с некоторой начальной точки $\bar{x}_0$ такое число раз, пока не будет выполнено условие $q(\bar{x}_{k+1}) \ge q(\bar{x}_k)$. С этого момента для того, чтобы как можно ближе подойти к минимуму, величина

шага $\alpha_k$ уменьшается, итерационная процедура (3.25) повторяется и т. д.

Второй способ – это так называемый метод наискорейшего спуска. Здесь на каждой итерации величина $\alpha_k$ выбирается таким образом, чтобы в направлении спуска целевая функция достигала своего минимального значения. Другими словами, $\alpha_k$ выбирается из условия

$$q(\bar{x}_k - \alpha_k \operatorname{grad} q(\bar{x}_k)) = \min_{\alpha_k \geq 0} q(\bar{x}_k - \alpha_k \operatorname{grad} q(\bar{x}_k)). \qquad (3.26)$$

Такой подход предполагает решать задачу скалярной (одномерной) минимизации (3.26) по переменной $\alpha_k$ на каждой итерации, что может оказаться весьма трудоемким, особенно в условиях, когда вычисления функции $q(\bar{x})$ сложны, либо она не дифференцируемая. В то же время, метод наискорейшего спуска дает выигрыш в числе машинных операций, так как обеспечивает движение с наиболее выгодной величиной шага $\alpha_k$.

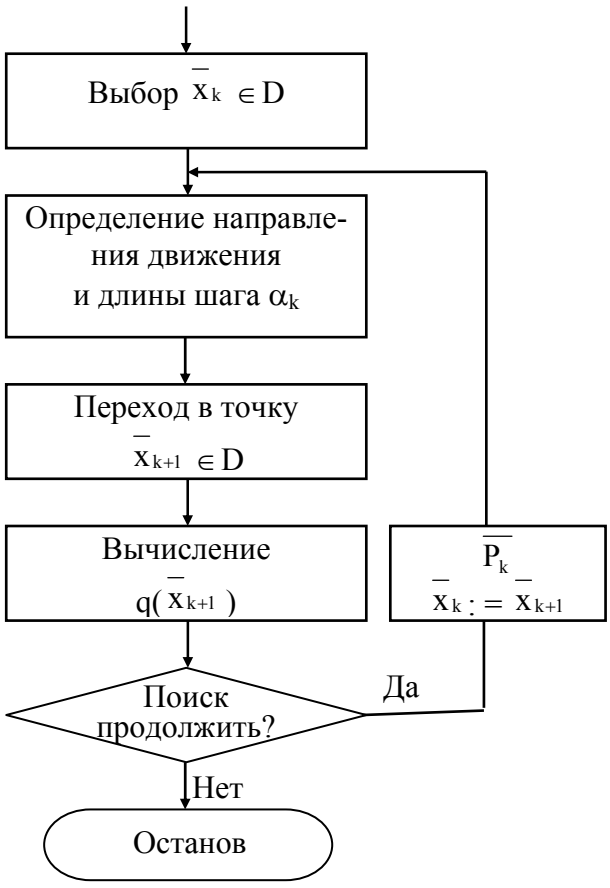


Рис. 3.7. *Схема одной итерации поиска: (: =) – знак присвоения*

Метод наискорейшего спуска отличается от обычного градиентного тем, что с выбором $\alpha_k$ по первому способу обеспечивает направление движения из точки $\bar{x}_k$ по касательной к линии уровня в точке $\bar{x}_{k+1}$, причем звенья такого зигзагообразного спуска ортогональны между собой.

В общем случае считается, что в рассмотренных выше методах градиентного спуска последовательность точек

$\{\bar{x}_k\}$_сходится к стационарной точке функции $q(\bar{x}^*)$. Можно доказать следующую теорему [Мухачева, Рубинштейн, 1977]: если функция $q(\bar{x})$ ограничена снизу, ее градиент удовлетворяет условию Липшица ($\left\|\text{gradq}(\bar{x}) - \text{gradq}(\bar{y})\right\| \le R\left\|\bar{x} - \bar{y}\right\|$, где R – константа Липшица $\forall x, y \in E_n$), и выбор $\alpha_k$ производится одним из описанных выше способов, то, какова бы ни была начальная точка $\bar{x}_0$ $\left\|\text{gradq}(\bar{x}_k)\right\| \to 0$ при $k \to \infty$.

Теорема дает практические условия прекращения итерационной процедуры поиска в виде

$$\left.\begin{array}{l} \left\|\text{gradq}(\bar{x}_k)\right\| \le \delta; \\ \text{либо} \\ \left\|\dfrac{\partial q}{\partial x_i}\right\|_{\bar{x}=\bar{x}_k} \le \delta, i = \overline{1,m,} \end{array}\right\} \qquad (3.27)$$

где $\delta > 0$ – число, характеризующее точность нахождения минимума.

Пример: $q(x_1, x_2, x_3) = 2x_1^2 + 5x_2^2 + 3x_3^2 \to \min$.

В качестве начальной точки возьмем $\bar{x}_0 = (3, 1, 1)$. Тогда $q(\bar{x}_0) = 26$.

grad $q(\bar{x}) = (\partial q/\partial x_1; \partial q/\partial x_2; \partial q/\partial x_3) = (4x_1, 10x_2, 6x_3)$.

Первая итерация. Вычислим градиент в точке $\bar{x}_0$: grad $q(\bar{x}_0) = (12, 10, 6)$, Сделаем шаг в направлении антиградиента из точки $\bar{x}_0$ в точку $\bar{x}_1$. Для этого используем формулы (3.19):

$$x_1^1 = x_1^0 - \alpha_k \left.\frac{\partial q}{\partial x_1}\right|_{\bar{x}=\bar{x}_0},$$

$$x_2^1 = x_2^0 - \alpha_k \left.\frac{\partial q}{\partial x_2}\right|_{\bar{x}=\bar{x}_0},$$

$$x_3^1 = x_3^0 - \alpha_k \left.\frac{\partial q}{\partial x_3}\right|_{\bar{x}=\bar{x}_0}.$$

В нашем случае

$$\left.\begin{aligned} x_1^1 &= 3 - 12\,\alpha_k, \\ x_2^1 &= 1 - 10\,\alpha_k, \\ x_3^1 &= 1 - 6\,\alpha_k. \end{aligned}\right\} \qquad (3.28)$$

Используя метод наискорейшего спуска, выберем длину шага $\alpha_k$ из условия:

$$\min_{\alpha_k \ge 0} q(\bar{x}_0 - \alpha_k \operatorname{grad} q(\bar{x}_0)) =$$

$$= \min_{\alpha_k \ge 0} (2(3 - 12\alpha_0)^2 + 5(1 - 10\alpha_0)^2 + 3(1 - 6\alpha_0)^2).$$

Для нахождения минимума по переменной $\alpha_0$ приравняем производную по этой переменной к нулю

$$48(3 - 12\alpha_0) - 100(1 - 10\alpha_0) - 36(1 - 6\alpha_0) = 0,$$

или после преобразований: $448\,\alpha_0 = 70$. Отсюда $\alpha_0 = 0{,}16$, учитывая соотношения (3.22),

$$\bar{x}_1 = (1{,}08; -0{,}6; 0{,}04).$$

Здесь

$$q(\bar{x}_1) = 4{,}14 < q(\bar{x}_0);\ \operatorname{grad} q(\bar{x}_1) = (4{,}32; -6; 0{,}24).$$

Заметим, что с условиями (3.22)

$$\left\|\frac{\partial q}{\partial x_i}\right\|_{\bar{x}=\bar{x}_1} < \left\|\frac{\partial q}{\partial x_i}\right\|_{\bar{x}=\bar{x}_0} \text{ и } \|\operatorname{grad}\ q(x_1)\| < \|\operatorname{grad}\ q(x_0)\|.$$

Вторая итерация. Вновь по формулам (3.20) рассчитываем

$$x_1^2 = x_1^1 - \alpha_1 \left.\frac{\partial q}{\partial x_1}\right|_{\bar{x}=\bar{x}_1},$$

$$x_2^2 = x_2^1 - \alpha_1 \left.\frac{\partial q}{\partial x_2}\right|_{\bar{x}=\bar{x}_1}, \quad \text{или} \quad \left.\begin{array}{l} x_1^2 = 1{,}08 - 4{,}32\alpha_1 \\ x_1^2 = -0{,}6 + 6{,}00\alpha_1 \\ x_1^2 = 0{,}04 - 0{,}24\alpha_1 \end{array}\right\}.$$

$$x_3^2 = x_3^1 - \alpha_1 \left.\frac{\partial q}{\partial x_3}\right|_{\bar{x}=\bar{x}_1},$$

Величину $\alpha_1$ выбираем из условия

$$\min_{\alpha_1 \geq 0} q(\bar{x}_1 - \alpha_1 \operatorname{grad} q(\bar{x}_1)) =$$

$$= \min_{\alpha_1 \geq 0} (2(1{,}08 - 4{,}32\alpha_1)^2 + 5(-0{,}6 + 6\alpha_1)^2 + 3(0{,}04 - 0{,}24\alpha_1)^2).$$

Приравнивая производную по $\alpha_1$ к нулю и решая уравнение, получим $\alpha_1 = -0{,}13$. Тогда

$$\bar{x}_2 = (0{,}52;\ 0{,}18;\ 0{,}01);\ q(\bar{x}_2) = 0{,}7 < q(\bar{x}_1) = 4{,}14 < q(\bar{x}_0) = 26.$$

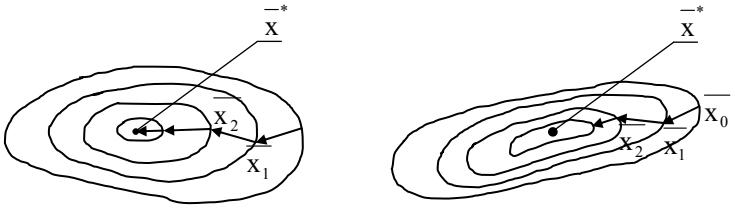


Рис. 3.8. *Траектории поиска методом наискорейшего спуска*

Как видим, с каждой итерацией $q(\bar{x}_k)$ приближается к своему оптимальному значению равному 0.

Можно показать, что градиентные методы имеют сходимость со скоростью геометрической прогрессии. На рис. 3.8 видно, что чем ближе линии уровня функции $q(\bar{x})$ к окружности, тем быстрее сходятся градиентные методы. В тех случаях, когда поверхности уровня сильно вытянуты (функция имеет так называемый «овражный» характер), этот метод может сходиться очень медленно и не обеспечить заданной точности поиска.

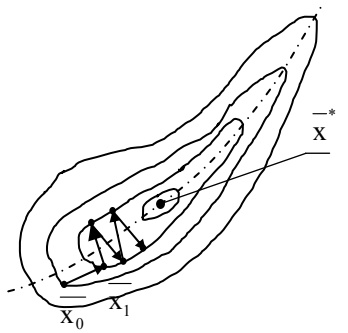


Рис. 3.9. *Градиентный метод поиска минимума овражной функции*

Например, если точка минимума лежит на дне оврага (на узком гребне – если решается задача на максимум), то метод может, вызвать прыжки через овраг (рис. 3.9). Для ускорения поиска экстремума в этом случае используется так называемый овражный метод. Опишем простейший его вариант.

Из двух начальных точек $\bar{x}_0$ и $\bar{x}_1$ производится одним из градиентных методов спуск в точки $\bar{x}_{00B}$ и $\bar{x}_{10B}$ на дне оврага. Допустим, что при этом $q(\bar{x}_{00B}) < q(\bar{x}_{10B})$. Далее делается движение по дну оврага, для этого полагается

$$\bar{x}_2 = \bar{x}_{10B} + \frac{\bar{x}_{10B} - \bar{x}_{00B}}{\left|\bar{x}_{10B} - \bar{x}_{00B}\right|} h$$

где $h > 0$ – называется овражным шагом.

Из точки $\bar{x}_2$, которая, вообще говоря, находится на «склоне оврага», производится спуск на дно оврага в точку $\bar{x}_{20B}$, определяется точка $\bar{x}_3$ и т.д. В общем, процесс движения по дну оврага описывается следующей итерационной формулой:

$$\bar{x}_{k+1} = \bar{x}_{k0B} + \frac{\bar{x}_{k0B} - \bar{x}_{k-10B}}{\left|\bar{x}_{k0B} - \bar{x}_{k-10B}\right|} h . \qquad (3.29)$$

Если уже найдены $\bar{x}_{00B}$, $\bar{x}_{10B}$,..., $\bar{x}_{k0B}$, то из точки $\bar{x}_{k+1}$ осуществляется спуск и находится следующая точка $\bar{x}_{k+10B}$ на дне оврага (рис. 3.10). Величина h на каждой итерации поиска подбирается эмпирически в зависимости от информации о поведении минимизируемой функции. При этом величину овражного шага следует выбирать так, чтобы: а) по возможности быстрее проходить прямолинейные участки на «дне оврага» за счет увеличения овражного шага; б) на крутых поворотах за счет уменьшения шага

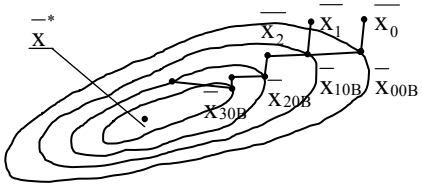


Рис. 3.10. *Овражный поиск минимума (спуск из точек $\bar{x}_k$ в $\bar{x}_{k0B}$ схематично изображен в виде отрезка прямой)*

избежать выброса из «оврага»; в) добиться меньшего отклонения точек $\bar{x}_{k0в}$ от «дна оврага», и тем самым сократить объем вычислений в процедурах градиентного спуска.

*Учет ограничений.* Ограничения в задаче нелинейного программирования создают дополнительные трудности ее решения. Будем рассматривать ограничения в виде неравенств $h_k(\bar{x}) \le 0$, $k = \overline{1,P}$.

Заметим, что для достижения оптимума не обязательно двигаться в направлении градиента, а можно осуществлять поиск по любому направлению, вдоль которого функция $q(\bar{x})$ убывает. Поэтому, если в процессе движения к оптимуму встретилось ограничение (граница) допустимой области решений D, поиск можно продолжать вдоль этой границы. Одним из простейших способов такого движения является зигзагообразное движение [Альянах, 1988]. В этом случае при нарушении ограничений оптимизацию следует проводить не по функции $q(\bar{x})$, а по k нарушенным ограничениям, минимизируя сумму $q(\bar{x}) = \sum_k h_k(\bar{x})$ $(k \le P)$. Определяя антиградиент этой суммы – $\mathrm{grad}\, q(\bar{x})$, организуют движение в направлении этого антиградиента, тем самым, производя возврат в область допустимых решений D.

Такой подход эквивалентен минимизации некоторой функции $Q(\bar{x})$ следующего вида: $Q(\bar{x}) = \begin{cases} q(\bar{x}) \text{ при } h_k(\bar{x}) \le 0,\ k = \overline{1,P}, \\ \sum_k h_k(\bar{x}) \text{ при } h_k(\bar{x}) > 0,\ k \le P. \end{cases}$

На рис. 3. 11 показан пример такого поиска минимума.

***Случайный поиск с локальной оптимизацией.*** Описанные выше градиентные процедуры в условиях многоэкстремальности целевой функции $q(\bar{x})$ приводят лишь к локальному минимуму, в районе которого была выбрана начальная точка $\bar{x}_0$ $(\bar{x}_0 \in \delta_\rho \subset D)$ – область протяжения локального минимума. С целью улучшения результатов поиска можно предложить провести градиентный

спуск с новой начальной точки $\bar{x}_1 \in D$ ($\bar{x}_1 \neq \bar{x}_0$), выбрав в качестве окончательного решения наименьший из двух локальных минимумов и т.д. Предположив теперь, что начальные точки для локальных спусков выбираются в соответствии с некоторым случайным механизмом (например, по равномерно закону распределения), получим так называемый случайный поиск с локальной оптимизацией.

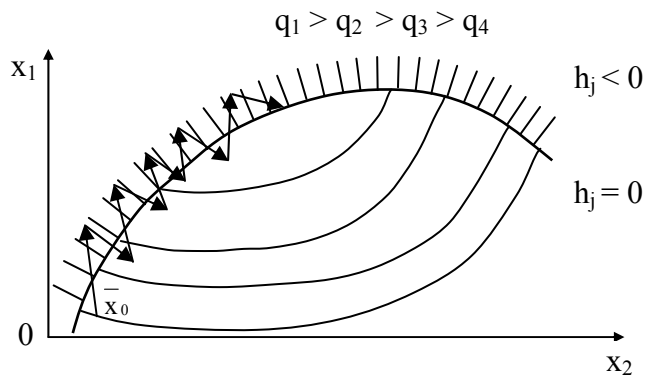


Рис. 3. 11. *Градиентный метод в условиях ограничений*

Рассмотрим более подробно процедуру такого поиска. Задается начальная случайная точка $\bar{x}_0 = (x_1^0, x_2^0, \ldots, x_m^0) \in D$. Далее одним из методов локальной оптимизации (например, градиентным спуском) производится поиск локального минимума $q_0 = q(\bar{x}_0^*)$. Значения $\bar{x}_0^*$ и $q_0$ запоминаются. Выбирается новая случайная точка $\bar{x}_1 = (x_1^1, x_2^1, \ldots, x_m^1) \in D$. Снова производится поиск локального минимума $q_1 = q(\bar{x}_1^*)$. Если очередной результат лучше предыдущего в смысле целевой функции ($q_1 < q_0$), то запоминаются соответствующие значения $\bar{x}_1^*$ и $q_1$. В противном случае ($q_0 < q_1$) остаются прежние значения $\bar{x}_0^*$ и $q_0$ – попытка считается безуспешной. Поиск прекращается после того, как совершится $\mu$ безуспешных попыток улучшить результат.

Если количество локальных экстремумов заранее неизвестно, то по заданному значению вероятности нахождения глобального экстремума можно с достаточной точностью оценить целое число $\mu$. Пусть есть $\beta$ минимумов из них $\beta - 1$ принадлежит области $\delta_\rho \subset D$ ($\rho = 1, 2, .., \beta - 1$), а один – глобальный области $\delta_\beta \subset D$. Производится моделирование равномерно распределенной в об-

ласти D случайной точки μ раз. Обозначим через P вероятность попадания точки в область притяжения $\delta_\rho$, а через αP ($0 < \alpha < 1$) – вероятность попадания точки в область притяжения $\delta_\beta$. Тогда, очевидно, $(\beta - 1)P + \alpha P = 1$, а вероятность пропуска глобального экстремума

$$\bar{P} = (1 - \alpha P)^{\mu} = \left(1 - \frac{\alpha}{\beta - 1 + \alpha}\right)^{\mu} = \left(\frac{\beta - 1}{\beta - 1 + \alpha}\right)^{\mu} \qquad (3.30)$$

Задавая в формуле (3.24) $\bar{P}$, всегда можно оценить целое число μ. Если количество локальных экстремумов заранее неизвестно, то целое число μ можно оценить экспериментальным путем. Очевидно, что по мере приближения к области минимума число следующих подряд реализаций μ будет возрастать. Количество неудачных реализаций в точке оптимума, естественно, становится бесконечно большим, независимо от того, где находится эта точка – на границе области или нет. Функциональная зависимость между критерием останова и относительной погрешностью δq устанавливается экспериментально.

## 3.4. Генетические алгоритмы

Общая идея построения генетических алгоритмов сводится к имитации процесса функционирования генных структур в процессе эволюции живых организмов.

Стандартный генетический алгоритм, был впервые подробно описан и исследован в работе де Джонга [De Jong, 1975]. Он проанализировал простую схему кодирования генов битовыми строками фиксированной длины и соответствующих генетических операторов, выполнил значительное количество численных экспериментов, сравнивая результаты с оценками, предсказываемыми теоремой Холланда [Holland, 1969, 1975].

Как показали последующие исследования, данный алгоритм оказался эффективным не только при исследовании биологиче-

ских процессов, но и при построении оптимизационных моделей в самых разнообразных проблемных областях.

Канонический генетический алгоритм характеризуется следующими особенностями.

1. Задается функция оптимальности f(x), определяющая эффективность каждого найденного решения (найденной комбинации признаков). Формируемое решение кодируется как вектор x, который называется «хромосома» и соответствует битовой маске, то есть двоичному представлению набора исходных переменных. В хромосоме выделяются части вектора – «гены», изменяющие свои значения в определенных позициях – «аллелях».

2. В соответствии с определенными ограничениями инициализируется исходная «популяция» $P^0(x_1^0, x_2^0, ..., x_\lambda^0)$ потенциальных решений – совокупность решений на конкретной итерации, состоящая из некоторого количества хромосом $\lambda$, число которых задаётся изначально и в процессе перебора обычно не изменяется.

3. Каждая хромосома $x_i$, $i = 1,..., \lambda$ в популяции декодируется в форму, необходимую для последующей оценки, и ей присваивается значение эффективности $f(x_i)$ в соответствии с вычисленной функцией оптимальности. Кроме того, каждой хромосоме присваивается вероятность воспроизведения $P(x_i)$, $i = 1,..., \lambda$, которая зависит от эффективности данной хромосомы. Существуют различные схемы отбора, самая популярная из них – пропорциональный отбор:

$$p(x_i^t) = \frac{f(x_i^t)}{\sum_{j=1}^{\lambda} f(x_j^t)}. \qquad (3.31)$$

4. В соответствии с вероятностями воспроизведения $P(x_i)$ создается новая популяция хромосом, причем с большей вероятностью воспроизводятся наиболее эффективные элементы. Хромосомы производят потомков, используя операции рекомбинации:

кроссинговер (хромосомы скрещиваются, обмениваясь частями строк (рис. 3.12.)) и мутация (вероятностное изменение аллелей (рис. 3.13)).

5. Процесс останавливается, если получено удовлетворитель-

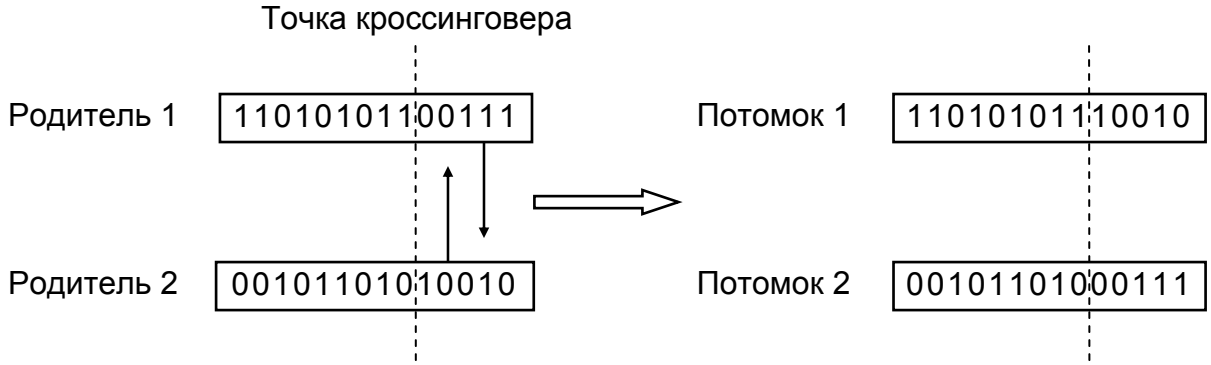


Рис. 3.12. *Одноточечная схема кроссинговера*

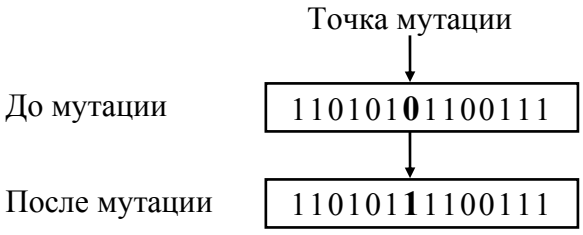


Рис. 3.13. *Мутация*

ное решение, либо исчерпано отведенное на эволюцию время. Если процесс не окончен, то вновь повторяются процессы оценки и воспроизведения новой популяции.

Операция воспроизведения на шаге 4 служит для создания следующей популяции на основе предыдущей при помощи операторов кроссинговера и мутации, которые имеют случайный характер. Каждой хромосоме промежуточной популяции $X_i^t$ в случае необходимости подбирается партнер и созданная хромосома помещается в результирующую популяцию.

Оператор кроссинговера производит скрещивание хромосом и обмен генетическим материалом между родителями для получения потомков. Этот оператор служит для исследования новых областей пространства и улучшения существующих (эволюционное приспособление). Простейший одноточечный кроссинговер производит обмен частями, на которые хромосома разбивается точкой кроссинговера, выбираемой случайно.

Двухточечный кроссинговер обменивает кусок строки, попавшей между двумя точками. Предельным случаем является равномерный кроссинговер, в результате которого все биты хромосом обмениваются с некоторой вероятностью. Оператор мутации применяется к каждому биту хромосомы с небольшой вероятностью ($p_i \approx 0{,}001$), в результате чего бит (аллель) изменяет значение на противоположное. Мутация нужна для расширения пространства поиска («эволюционное исследование») и предотвращения невосстановимой потери бит в аллелях.

Существует также оператор воспроизведения, называемый «инверсия», который заключается в реверсировании аллелей между двумя случайными позициями, однако для большинства задач он не имеет практического смысла и поэтому мало эффективен. При использовании генетических алгоритмов в задачах оптимизации простой рандомизированный перебор при поиске оптимума по методу Монте-Карло можно поменять на хранение популяции лучших с точки зрения функционала объектов и добавления новых объектов посредством мутации одного объекта, либо посредством кроссовера – обмена подобными частями у двух случайных объектов. Мутация – это аналог случайного шага в методе Монте-Карло, а кроссовер и популяция привлечены из биологии, где естественный отбор оптимизирует генотип подобным образом. Обобщенная блок-схема генетического алгоритма при его реализации в модели оптимизации представлена на рис. 3.14.

Для исследования эффективности генетических алгоритмов используется понятие «шаблон», который является некоторым представлением подмножества строк, имеющих одинаковые значения в специфицированных позициях (шаблоны гиперплоскости различной размерности в одномерном пространстве).

Шаблоны определяются с помощью элементов множества $\{0,1,*\}^L$ и для данного шаблона $H \in \{0,1,*\}^L$, где L – длина хромосомы в битах, символ «*» означает, что в данной позиции строка-

бит может принимать значение 1 или 0. Строка «а» является одним из представителей данного шаблона тогда и только тогда, когда биты в специфицированных позициях шаблона (то есть, биты, содержащие 0 или 1) идентичны соответствующим битам строки «а».

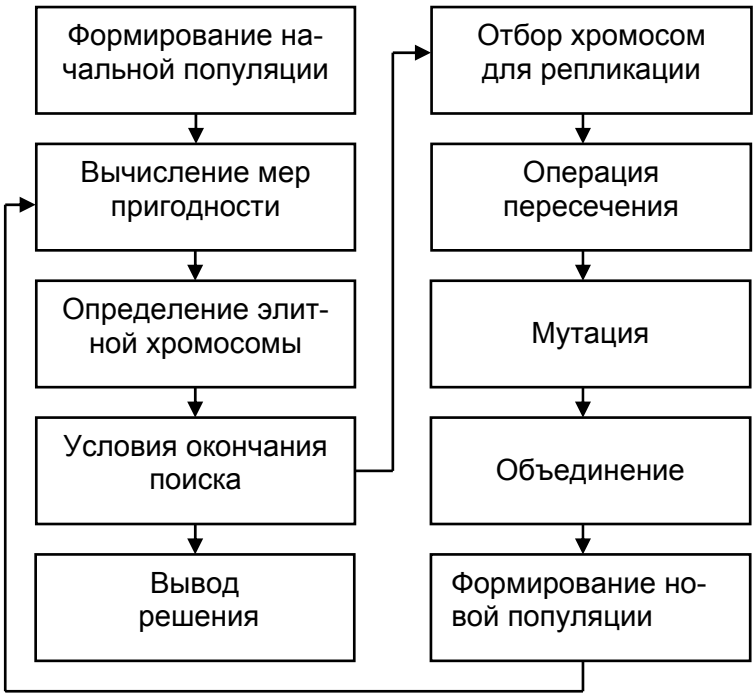


Рис. 3.14. *Обобщенная блок-схема генетического алгоритма*

Шаблоны представляют собой гиперплоскости различной размерности в L-мерном пространстве. Генетический алгоритм обрабатывает шаблоны, и производит выборку значительного числа гиперплоскостей из областей с высокой приспособленностью, причем в течение одного поколения популяции оценивается $O(\lambda^3)$ структур, хотя в реальности популяция содержит только L индивидуумов. Этот эффект носит название неявный параллелизм и отличает эволюционные процессы от различных эвристических и случайно-поисковых методов, в которых единственное решение развивается само по себе, а предыдущий опыт не используется.

Решение, получаемое при помощи генетического алгоритма, по своему характеру субоптимально, но это не мешает применять метод для поиска глобальных экстремумов в широком классе задач оптимизации многоэкстремальных функций.

Формально генетический алгоритм можно описать следующим образом:

$$ГА = (P^0, \lambda, L, s, p, f, t) \qquad (3.32)$$

где $P^0 = (a_1^0, a_2^0, ..., a_\lambda^0)$ – исходная популяция, $a_i^0$ – решение задачи, представленное в виде хромосомы; $\lambda$ – целое число (размер популяции); L – целое число (длина хромосомы популяции); s – оператор отбора; p – отображение, определяющее рекомбинацию (кроссинговер, мутация); f – функция оптимальности; t – критерий остановки.

Идею работы генетического алгоритма проиллюстрируем с помощью следующего примера. Пусть необходимо найти максимум некоторой функции одной переменной, показанной на рисунке 3.15.

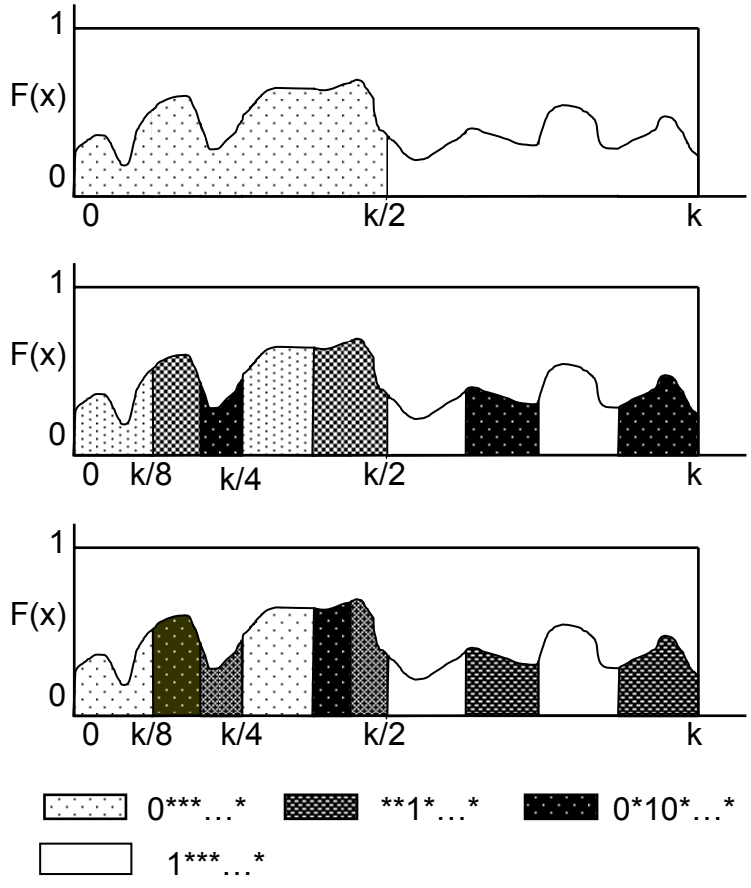


Рис. 3.15. Поиск оптимума функции с помощью генетического алгоритма

Гиперплоскость 0***...* покрывает первую половину пространства поиска, гиперплоскость 1***...* – вторую половину пространства поиска. Как видно из рисунка, оптимальность строк, принадлежащих 0***...*, в среднем выше, чем оптимальность строк, принадлежащих 1***...*. Поэтому было бы желательно иметь больше представителей первой гиперплоскости. Следовательно, отбор должен отдавать предпочтение таким строкам как 0***...*. На втором рисунке показана подобласть, соответствующая **1 **...* и пересечение **1**...* и 0***...* – подобласть 0*1**...*. Третий рисунок показывает подобласть 0*10*...*. Таким образом,

видно, что выбор определенных подобластей пространства поиска может осуществляться независимо от наличия локальных минимумов. Одновременно, увеличение количества точек, соответствующих подобластям с относительно большей эффективностью, не может гарантировать сходимости к глобальному оптимуму. Тем не менее, всегда возможно найти решение, которое будет относительно эффективно по сравнению с альтернативными.

Наиболее часто используется бинарная кодировка для представления элемента популяции, выбор которой определяется тем, что двоичный алфавит позволяет обрабатывать максимальное количество информации по сравнению с другими схемами кодирования. Существуют альтернативные двоичному кодированию схемы, например, использование представления с плавающей точкой. В зависимости от свойств конкретной решаемой задачи обе схемы кодирования имеют свои достоинства и недостатки.

Оператор отбора s порождает промежуточную популяцию $P^t$ из популяции $P^t$ посредством отбора и генерации новых копий элементов $P^t$: $P^t = s(P^t)$. Функция оптимальности f, обеспечивающая обратную связь от результатов оптимизации в течение поколения t, используется для отбора конкурентоспособных индивидуумов популяции.

При отборе часто используется ранг элементов популяции (rank) который задается следующим образом:

$$\forall i \in \{1,..., \lambda\}: \text{rank}(a^t_i) = i, \quad (3.33)$$
$$\text{если для } \forall j \in \{1,..., \lambda-1\}: f(a^t_j) \; o \; f(a^t_{j+1}),$$

где o – означает ≤ в случае минимизации и ≥ в случае максимизации.

Следовательно, можно использовать индекс i для обозначения ранга элемента. В соответствии с этим предполагается, что элементы сортируются согласно их конкурентоспособности (оптимальности) так, что $a^t_1$ будет лучшим элементом $P^t$.

Отбор производится на основании вероятностей $p_s(a^t_i)$, вы-

численных для каждого индивидуума популяции. В настоящее время применяются следующие основные схемы отбора:

пропорциональный отбор

$$p_s\left(a_i^t\right) = f\left(a_i^t\right) \Big/ \sum_{j=1}^{\lambda} f\left(a_j^t\right); \tag{3.34}$$

линейное ранжирование

$$p_s\left(a_i^t\right) = \frac{1}{\lambda}\left(\eta_{max} - \left(\eta_{max} - \eta_{min}\right)\frac{i-1}{\lambda-1}\right), \tag{3.35}$$

где $\eta_{min} = 2 - \eta_{max}$ и $\leq \eta_{max} \leq 2$;.

$(\mu, \lambda)$-равномерное ранжирование

$$p_s\left(a_i^t\right) = \begin{cases} 1/\mu, 1 \leq i < \mu; \\ 0 \quad , \mu \leq i \leq \lambda. \end{cases} \tag{3.36}$$

Наиболее часто используется пропорциональный отбор. Однако его использование может привести к тому, что простое добавление очень большой константы к целевой функции может уничтожить отбор, приводя к простому случайному выбору. Для устранения этого недостатка используется либо масштабирование оценок оптимальности родителей относительно наименьшего значения оптимальности в популяции, либо ранжирование в пропорциональной схеме отбора.

Еще одна из математических проблем, связанных с пропорциональным отбором состоит в том, что данная процедура не может гарантировать асимптотическую сходимость к глобальному оптимуму. Наилучшая хромосома в популяции может быть потеряна в любом поколения, и нет какой-либо гарантии, что результаты эволюции, достигнутые в ряде поколений, не будут утрачены. Одним из способов преодоления этого является использование так называемого элитного отбора, который всегда сохраняет наилучшую хромосому популяции. Этот вид отбора асимптотически сходится, но скорость сходимости может существенно различаться в зависимости от вида конкретной решаемой задачи.

Анализ пропорционального отбора позволил получить теоре-

му Холланда. Пусть $H^t$ обозначает определенный шаблон, который присутствует в популяции $P^t$ и $m(H^t)$ – количество индивидуумов популяции $P^t$, которые являются элементами $H^t$. Тогда

$$m(H^{t+1}) = m(H^t)f(H^t)/\ \overline{f^t} \tag{3.37}$$

$$\overline{f^t} = \frac{1}{\lambda}\sum_{i=1}^{\lambda} f\left(a_i^t\right) \tag{3.38}$$

$$f(H^t) = \frac{1}{m(H^t)}\sum_{\substack{i=1 \\ a_i^t \in H^t}}^{m(H^t)} f\left(a_i^t\right), \tag{3.39}$$

где $f^t$ – средняя оптимальность $P^t$, $f(H^t)$ –средняя оптимальность индивидуумов, являющихся элементами $H^t$.

Если предположить, что средняя оптимальность $H^t$ выше средней по популяции, то есть $f(H^t) = f^t + cf^t$, где c =const > 0, то в результате имеем $m(H^{t+1}) = m(H^t)\ (1 + c)$, и после t поколений, начиная с t = 0:

$$m(H^{t+1}) = m(H^0)\ (1 + c)^t. \tag{3.40}$$

Из (13.40) видно, что с течением времени число лучших индивидуумов популяции экспоненциально возрастает, а число индивидуумов, имеющих оптимальность ниже средней по популяции, экспоненциально уменьшается. Это остается верным и в случае учета эффектов, производимых генетическими операторами. Данный результат рассматривается как утверждение о том, что генетический алгоритм – оптимальная процедура для получения лучших элементов.

Если включить в анализ кроссинговер и мутацию (наиболее часто используемые генетические операторы), то необходимо добавить две новые характеристики: 1) порядок $\sigma(H)$ шаблона $H \in \{0,1,*\}^1$, определяемый количеством его зафиксированных позиций, то есть позиций, содержащих либо 1, либо 0; 2) длину $\delta(H)$ шаблона $H \in \{0,1,*\}^1$ – дистанцию между первой и последней зафиксированной позициями шаблона.

Пусть вероятность осуществления одноточечного кроссинго-

вера равна $P_c$, а вероятность мутации – $P_m$. В итоге, используя сделанные предположения, получаем следующее утверждение.

Теорема Холланда о шаблоне

$$m(H^{t+1}) \geq m(H^t)\,\frac{f(H^t)}{\overline{f^t}}\left(1 - p_c\,\frac{\delta(H^t)}{l-1} - \sigma(H^t)p_m\right), \quad (3.41)$$

означает, что хромосомы (строки), короткие (по отношению к кроссинговеру), низшего порядка (по отношению к мутациям), имеющие оптимальность выше средней по популяции, экспоненциально увеличивают свое представительство в последовательности поколений.

Для предельного случая (популяция бесконечного размера, пространство поиска непрерывно, циклически применяется пропорциональный отбор) доказана следующая теорема [Xiaofeng Q, Palmieti F, 1994]. Пусть выполняются условия: (1) неотрицательная функция g(x) определена на ограниченном подмножестве $F \in R^1$ и имеет единственный глобальный максимум при $x = x^* \in F$ $g^* = g(x^*)$; (2) существует пространственно-односвязная окрестность точки $x^*$, внутри которой g(x) непрерывна; (3) начальная плотность $f_0$ распределения ненулевая в точке $x^*$. Тогда последовательность $f_{k+1}(x) = \dfrac{f_k(x)g(x)}{\int\limits_F f_k(x)g(x)dx}$, $k = 0, 1, 2,\ldots$ сходится к $\delta(x-x^*)$ при $k \to \infty$, где $\delta(x-x^*)$ – функция Дирака.

После завершения отбора, выполняются генетические операции: кроссинговер и мутация. Обе операции имеют случайный характер (вероятность применения, выбор локуса внутри хромосомы). Соответственно элементу $a_i^t \in P_i^t$ выбирается партнер из $P^t$ для рекомбинации, если это необходимо (например, для кроссинговера) и строится новая хромосома. Операторы, приводящие непосредственно после своего завершения более чем к одному потомку, случайно или, используя какую-либо иную стратегию, отбирают одного или более элементов-потомков, которые будут

окончательным результатом операции, остальные элементы отбрасываются.

Однако возникает вопрос: существует ли какая-либо необходимость в мутации и кроссинговере, если согласно последней теореме, циклическое повторение отбора приводит к сходимости к оптимуму? При ответе на этот вопрос необходимо помнить, что при выводе теоремы использовалось предположение о популяции очень большого размера. Это подразумевает, что все пространство поиска покрыто элементами популяции с ненулевой вероятностью. Таким образом, оптимальное решение уже содержится в выборке и поэтому достаточно одного отбора для отыскания оптимальной точки или точек. В условиях популяции ограниченного размера мутация и кроссинговер являются теми операторами, которые позволяют исследовать области пространства поиска, не охваченные популяцией, и, которые могут содержать лучшие решения.

Традиционный и наиболее важный оператор рекомбинации, используемый генетическими алгоритмами, одноточечный кроссинговер, который осуществляется с вероятностью $P_c$ (часто $P_c$ = 0.6). Кроссинговер выполняется следующим образом: случайный выбор партнера для скрещивания $a_2 = (a_{2,1}...a_{2,L})$ из $P^t$; случайный выбор точки кроссинговера $x \in \{1,..., L - 1\}$; формирование двух новых индивидуумов $a_1' = \{a_{1,1}...a_{1,x}\ a_{2,x+1}...a_{2,L}\}$ и $a_2' = \{a_{2,1}...a_{2,x}\ a_{1,x+1}...a_{1,L}\}$; случайный выбор $a \in \{a_1', a_2'\}$.

Отсюда может быть подсчитан вклад, вносимый кроссинговером, в теорему Холланда. Доступно L – 1 точек для кроссинговера внутри строки, следовательно, вероятность того, что шаблон H будет разрушен кроссинговером, равна $\delta(H)/(L - 1)$. Так как кроссинговер носит вероятностный характер, данное выражение необходимо умножить на $P_c$.

Естественным развитием одноточечного кроссинговера являются схемы с несколькими точками. Предельным случаем явля-

ется равномерный кроссинговер, когда каждая пара битов внутри двух хромосом обменивается в соответствии с некоторой вероятностью. Несмотря на возможные различия в реализации всех вариантов, все типы кроссинговера обладают общим свойством: они контролируют баланс между дальнейшим использованием уже найденных хороших подобластей пространства и исследованием новых подобластей. Достигается это за счет неразрушения общих блоков внутри хромосом-родителей, сохраняющем «хорошие» паттерны, и одновременном исследовании новых областей в результате обмена частями строк (хромосом). Именно это наблюдение привело Холланда к идее введения концепции шаблона, которая использовалась при получении теоремы о шаблоне. Используя те же аргументы, Голдберг предложил гипотезу о строительном блоке: при условии использования коротких (низкого порядка) шаблонов генетический алгоритм сходится к гиперплоскости с наилучшем значением средней оптимальности, где короткие (низкого порядка) шаблоны интерпретируются как паттерны строк, которые в результате их компактной упаковки, достаточно невосприимчивы к потенциальным эффектам разрушения, вызванным кроссинговером. Совместное использование отбора и кроссинговера приводит к тому, что области пространства, обладающие лучшей средней оптимальностью, содержат больше элементов популяции, чем другие. Таким образом, эволюция популяции направляется к областям, содержащим оптимум с большей вероятностью, чем другие.

Мутации вносят новизну и предотвращают невосстановимую потерю аллелей в определенных позициях, которые не могут быть восстановлены кроссинговером, ограничивая тем самым преждевременное сжатие пространства поиска. Мутация представляет собой случайное изменение бита, вероятность которого довольно низкая ($P_m \approx 0.001$). Мутация функционирует следующим образом: 1) случайный выбор (с некоторой фиксированной вероятностью $P_m$) позиций $\{x_1 ... x_k) \subseteq \{1,..., L\}$ внутри битовой строки, под-

верженных мутации; 2) $a = (a_{1,1}...a_{1,x1-1}a'_{x1}a_{1,x1+1}...a_{1,xh-1}a'_{xh}a_{1,xh+1}...a_{1Ll})$, где $a'_{xi} \in \{0,1\}\ \forall i = 1,..., h$ выбираются случайным образом.

При использовании мутации вероятность того, что шаблон порядка $\sigma(H)$ будет разрушен, равна $1 - (1 - P_m)^{\sigma(H)}$. Для малых $P_m$ это приблизительно равно $\sigma(H)P_m$ , что объясняет наличие последнего члена в выражении, используемом в теореме о шаблоне.

В общем случае мутация может рассматриваться как процесс перехода между различными состояниями пространства поиска и характеризоваться условной плотностью вероятности нового состояния при данном старом. Следующая теорема [Xiaofeng Q, Palmieti F, 1994] показывает, что циклическое применение последовательности отбор-мутация направляет эволюцию элементов популяции к наиболее хорошим точкам пространства поиска.

Теорема. Пусть функция $g(x) \geq 0, \forall x \in F$ имеет глобальный максимум $g^*$ (возможно, во многих точках); существует непрерывная и пространственно-односвязная окрестность вокруг каждой точки глобального оптимума; исходная плотность распределения ненулевая, по крайней мере, в одной из таких окрестностей. Тогда существует последовательность условных плотностей распределения оператора мутации $\{f_{x_{k+1}|x_k}\}_{k=0}^{\infty}$ с матрицами ковариации $\{\sigma_k^2 I\}_{k=0}^{\infty}$ такая, что последовательность плотностей распределения внутри популяции, определенная как

$$f_{k+1}(x) = \frac{[f_k g] * f_{x_{k+1}|x_k}}{\int_F f_k g dx}, \qquad (3.42)$$

приводит к увеличению средней оптимальности и сходится к глобальному максимуму, то есть $E[g_{k+1}] \geq E[g_k]$ и $\lim_{k\to\infty} E[g_k] = g^*$.

Переход от поколения t к поколению t + 1 формально можно записать следующим образом:

$$s(P^t) = P^t = (a_1^t,..., a_\lambda^t),\ P^{t+1} = \rho\left(a_1^t,..., a_\lambda^t\right). \qquad (3.43)$$

Таким образом, отбор производит селекцию наиболее хороших элементов. Последующие кроссинговер и мутация накладывают стохастический шум на процесс эволюции, осуществляя переход и исследование новых областей и точек пространства.

При выполнении некоторых специальных условий генетический алгоритм способен отыскивать глобальный оптимум, что подтверждает следующая теорема [Harti, 1990]. Пусть выполняются условия:

1) Последовательность популяций $P^0$, $P^1$,... генерируемая алгоритмом – монотонна, то есть: $\forall i \in N$: $\min\{f(a) | a \in P^{t+1}\} \le \min\{f(a) | a \in P^t\}$.

2) $\forall a, a^*$ элемент $a^*$ достижим из a посредством мутации и кроссинговера, то есть, через последовательность переходов в ряде структур.

Тогда глобальный оптимум функции f отыскивается с вероятностью единица: $\lim_{t\to\infty} p\{a^* \in P^t\} = 1$.

Эффективность элемента популяции рассчитывается на основе функции оптимальности, которая включает в себя обычно несколько компонент (таких как, декодирование, учет ограничений, масштабирование и собственно целевая функция).

Для n-мерной с непрерывными параметрами оптимизационной задачи предполагается, что индивидуум популяции a состоит из $L = nL_x$, битов, где $L_x$ – количество бит, используемое для кодирования одной непрерывной переменной $x_i \in [a_i, b_i]$. Для бинарной кодировки переменных декодирующая функция, $\Gamma_{a,b,L}: \{0,1\}^L$, принимает форму:

$$\Gamma_{a,b,L}(\alpha_1,\ldots,\alpha_L) = a + \frac{b-a}{2^L - 1}\left(\sum_{i=1}^{L} \alpha_i 2^{i-1}\right). \qquad (3.44)$$

Кроме обычной двоичной кодировки часто используется код Грея, который в некоторых случаях позволяет увеличить эффективность генетического алгоритма. Для кода Грея декодирующая функция Г может быть, например, представлена в виде

$$\Gamma_{a,b,l}(\alpha_1,\ldots,\alpha_l) = a + \frac{b-a}{2^L-1}\left(\sum_{i=1}^{L}\left(\bigoplus_{j=1}^{i}\alpha_j\right)2^{i-1}\right), \qquad (3.45)$$

где $\oplus$ – суммирование по модулю 2.

Отличительной характеристикой кода Грея является то, что соседние целые числа отличаются друг от друга только на один бит.

Учет дополнительных ограничений, возможен, если использовать один из следующих методов.

1) Разработка специализированных генетических операторов таким образом, чтобы производимые с их помощью новые индивидуумы популяции подчинялись налагаемым ограничениям.

2) Использование штрафных функций для наказания некорректных индивидуумов в соответствии с их степенью нарушения ограничений.

Масштабирование предназначено для предотвращения доминирования какого-либо элемента популяции над остальными на ранних стадиях эволюции и для расширения диапазона значений оптимальности на финальных стадиях эволюции, когда разнообразие в популяции существенно снижается. Среди данного класса методов выделяется динамическое параметрическое перекодирование, которое реализуется следующим образом: когда наблюдается сходимость популяции, максимальные и минимальные значения диапазона пересчитываются для меньшего интервала (окна), и процесс итерируется далее. Этим достигается динамическое изменение диапазона решений при приближении к оптимуму. Таким образом, масштабирование контролирует селективный отбор внутри популяции.

Возможны также другие методы масштабирования, такие как, например, линейное динамическое масштабирование, логарифмическое масштабирование, экспоненциальное масштабирование и др.

Критерий остановки эволюции t может быть определен про-

извольным образом (например, получение удовлетворительного решения, достижение определенного числа поколений, количество затраченного времени, низкая скорость сходимости и т.п.)

Из теоремы о шаблоне:

$$m(H^{t+1}) \geq m(H^t)\ \frac{f(H^t)}{\overline{f^t}}\left(1 - p_c\ \frac{\delta(H^t)}{l-1} - \sigma(H^t)p_m\right), \quad (3.46)$$

видно, что шаблоны порождают увеличивающееся или уменьшающееся число представителей в соответствии с их оптимальностью. Фактор роста $\gamma$ выражается следующим образом:

$$\gamma = \frac{f(H^t)}{\overline{f^t}}\left(1 - p_c\ \frac{\delta(H^t)}{l-1} - \sigma(H^t)p_m\right). \quad (3.47)$$

Согласно данной теореме гарантируется рост числа шаблонов в последовательности поколений при соблюдении трех условий: 1) оптимальность шаблона выше средней по популяции; 2) его длина относительно мала; 3) он низкого порядка.

Когда все три условия выполняются, то данный шаблон называется строительным блоком. Таким образом, значение фактора роста (большее или меньшее 1) является указанием на то, является ли данный шаблон строительным блоком. Когда $\gamma > 1$, последующие поколения будут содержать увеличивающееся количество строк, содержащих данный шаблон. Новые строки будут создаваться посредством рекомбинации данного строительного блока и других строительных блоков.

Однако существуют ситуации, нарушающие сходимость алгоритма к оптимальным точкам. Предположим, что оптимальность каких-либо двух коротких, низкого порядка шаблонов выше средней, а оптимальность их объединения – ниже средней. Например, оптимальность 00***** и *****00 выше средней, в то время как оптимальность 00***00 значительно меньше оптимальности дополнения, 11***11, являющегося строительным блоком точки оптимума, 1111111. В этом случае генетический алгоритм сходится к точкам, имеющим меньшую оптимальность (напри-

мер, 0011100), потому, что с высокой степенью вероятности кроссинговер будет разрушать необходимые в данном случае комбинации (11***11). Данная ситуация показывает проявление так называемой проблемы ложного оптимума. Способ кодирования и упорядочивания битов внутри строки может направить алгоритм по ложному пути, вызывая сходимость к локальным оптимумам. Для преодоления данной проблемы были предложены два варианта: 1) использование предварительной информации для правильного упорядочивания битовых комбинаций внутри строки (сцепление битов); 2) использование инверсии.

Первый вариант является вполне приемлемым решением, но он предполагает наличие предварительной информации о свойствах оптимизируемой функции. В предыдущем примере, если бы мы имели такую информацию заранее, то мы закодировали бы строку так, чтобы четыре важных для нас бита были бы соседними (например, 1111***). В таком случае кроссинговер с меньшей бы вероятностью разрушал необходимую комбинацию битов, фактор роста был бы выше, и мог бы быть сформирован необходимый строительный блок. И хотя в некоторых случаях возможно идентифицировать необходимую битовую комбинацию, в общем случае требование такой информации является нежелательным и ограничивает спектр приложений алгоритма.

Если упорядочивание битов внутри строки при решении определенной задачи является случайным, то было бы полезным знать вероятность случайного кодирования группы битов порядка k с определяющей длиной $\sigma = d$ в строке длины L. Для такого варианта соответствующая плотность вероятности равна:

$$P(\delta = d) = (l - d)\frac{\binom{d-1}{k-2}}{\binom{l}{k}}, \qquad (3.48)$$

а кумулятивное распределение –

$$P(\delta = d) = \left[\frac{k(l-d+1)+d+1}{d+1}\right]\binom{d+1}{k}\Big/\binom{l}{k}. \qquad (3.49)$$

Математическое ожидание длины строительного блока:

$$\frac{\langle\delta\rangle}{l+1} = \frac{k-1}{k+1}. \qquad (3.50)$$

Рассматривая левую часть этого выражения как нормализованную длину, можно увидеть, что она быстро растет с увеличением k. Даже для блока третьего порядка математическое ожидание нормализованной определяющей длины составляет 0,5. Таким образом, даже для битовых комбинаций низкого порядка шансы быть закодированными с необходимым упорядочиванием (быть сцепленными) являются низкими. По этой причине (а также потому, что априорная информация не всегда доступна) была предложена инверсия и другие операторы, переупорядочивающие биты внутри строки во время работы алгоритма.

Инверсия нацелена на создание строительного блока, основываясь на предположении, что необходимое сцепление битов может быть найдено одновременно с поиском хороших (оптимальных) битов. Рассмотрим, как работает инверсия. Добавление инверсии требует идентификации каждого бита. Например, строка 1100011 могла бы быть идентифицирована следующим образом ((1 1) (2 1) (3 0) (4 0) (5 0) (6 1) (7 1)). Это позволяет, несмотря на возможное перемешивание битов внутри строки, интерпретировать ее смысл без проблем, поскольку каждый бит имеет свой идентификатор.

Стандартный оператор инверсии выбирает две позиции внутри строки и перебрасывает полученную подстроку между этими позициями. Пусть инверсия происходит между 2-м и 3-м битами и после 7-го бита. Тогда получаем ((1 1) (2 1) (71) (6 1) (5 0) (4 0) (3 0)).

Видно, что в данном случае инверсия получила необходимое сцепление битов для формирования строительного блока 1111***.

Этот пример показывает, что инверсия может быть применена для осуществления нужного сцепления битов. Однако вопрос о степени ее эффективности является спорным. Исследования показали невысокую эффективность инверсии. Инверсия играет по отношению к упорядочиванию битов ту же роль, что и мутация по отношению к аллелям. Оба оператора предотвращают преждевременное уменьшение разнообразия внутри популяции. Было найдено, что для того, чтобы инверсия обеспечивала сходимость к оптимуму, вероятность ее применения должна быть очень низкой. Но тогда возникает вопрос о ее практической ценности при одновременном поиске хороших аллелей и необходимого сцепления битов, так как время эволюции при ее применении значительно возрастает.

Для поиска возможного преодоления этой трудности можно попытаться использовать известные природные механизмы. Основываясь на них, был разработан мобильный генетический алгоритм, использующий строки переменной длины и специализированные генетические операторы для оперирования над такими строками. Об этом алгоритме речь пойдет ниже. Пока же отметим, что генетические алгоритмы относятся к классу методов оптимизации, обладающих наилучшими нелокальными свойствами и охватывающих (благодаря своей гибкости) широкий диапазон проблем.

Поскольку популяция каким-то образом рассеяна в пространстве объектных параметров $R^n$, у нее мало шансов сосредоточиться вокруг первого попавшегося локального минимума. Тем не менее, в силу рекурсивного характера алгоритма, это не может гарантировать глобальную сходимость. Возможность отыскания глобального оптимума среди множества локальных зависит от отношения размера популяции к числу объектных параметров, от природы множества оптимизации, адекватного выбора способа кодирования и упорядочивания переменных в виде элементов популяции, формы представления управляющих параметров генети-

ческого алгоритма, контролирующих отбор, кроссинговер, мутацию и возможные другие генетические операторы, например, инверсию. Всегда остается выбор между максимальной скоростью сходимости с одной стороны, и глобальной сходимостью с другой.

Возникает естественный вопрос: возможна ли разработка нового генетического алгоритма, имеющего больше общих свойств с реальными генетическими механизмами, и вследствие этого увеличивающего свою эффективность по сравнению со стандартным вариантом?

Ответом на этот вопрос была идея разработки нового типа генетических алгоритмов – мобильного генетического алгоритма [Goldberg, Korb, Deb, 1989]. Этот алгоритм основан на комбинации относительно коротких, протестированных блоков генов, направленной на создание больших генотипов, кодирующих все нужные характеристики решаемой задачи. Введение в структуру генетических алгоритмов генов и хромосом переменной длины и новых генетических операторов приводит к более хорошему решению по сравнению со схемами классического типа.

Мобильный генетический алгоритм имеет следующие основные отличия от стандартного генетического алгоритма:

1) Использование строк переменной длины, которые могут быть либо пере-, либо недоопределены по отношению к решаемой задаче.

2) Введение правил чтения или экспрессии генов.

3) Использование операторов CUT (разрезание) и SPLICE (*сплайсинг* – сцепление) вместо традиционной схемы кроссинговера, оперирующего над генами фиксированной длины.

4) Конкуренция между строительными блоками генотипа для отбора наиболее оптимальных.

5) Деление эволюции на две фазы: предварительная фаза и фаза процессинга.

Свое название данный тип алгоритма получил из-за исполь-

зования строк переменной длины. Кроме того, алгоритм снимает ограничение, связанное с фиксированным положением гена внутри хромосомы, свойственного стандартному генетическому алгоритму. Это достигается за счет определения гена как упорядоченной пары, идентифицирующей имя гена, его значение и определения хромосомы как объединения таких генов.

Например, рассмотрим случай с длиной L = 3 и соответствующую ему строку из трех битов 011 стандартного генетического алгоритма. Данная строка будет представлена в новом алгоритме (используя LISP-подобный синтаксис) как ((1 0) (2 1) (3 1)), где каждый бит идентифицируется его именем и значением. В этой строке первым объектом является ген "1" со значением 0, вторым – ген "2" со значением 1 и третьим – ген "3" со значением 1. Так как и имя, и значение гена определены, то может быть достигнута любая необходимая перестройка внутри хромосомы для сцепления определенных генов.

Такая же схема идентификации гена применяется и во многих других алгоритмах, использующих перемещаемые локусы внутри хромосомы. Но мобильный алгоритм имеет особенность, отличающую его от стандартной схемы: не накладывается никаких требований на наличие избыточного количества генов внутри хромосомы и на наличие одних и тех же генов с взаимно исключающими значениями. Подобные ситуации разрешаются за счет использования механизма экспрессии.

Переопределение устраняется за счет процедуры экспрессии генов, использующей правило «слева-направо», согласно которому строка сканируется слева направо, и ген, расположенный левее, будет считаться активным в данной хромосоме. Например, строка ((1 0) (2 1) (1 1)) за счет экспрессии перейдет в строку ((1 0) (2 1)) потому, что вторая версия первого гена не будет использована согласно применяемому правилу. Правило «слева-направо» было выбрано вместо различных схем, использующих понятие эффективности генов, так как при применении таких

схем гены, получившие большое преимущество в начале эволюции (большие значения оптимальности) могут заблокировать проявление генов, которые на данной стадии менее эффективны, но являются строительными блоками искомого оптимума.

Для устранения проблемы недоопределения используется другой подход. В некоторых задачах недоопределенность не является проблемой, так как любая структура любого размера может быть интерпретирована естественным образом. В задачах параметрической оптимизации с определенным числом параметров все переменные должны быть представлены в целевой функции для ее оптимизации.

В таких случаях мобильный генетический алгоритм заполняет недоопределенные позиции хромосомы участком наиболее конкурентоспособной хромосомы-предшественницы, которая является оптимальной на предшествующем уровне организации. Идея состоит в том, что наиболее хорошая (конкурентоспособная) структура, получаемая на одном уровне эволюции и приводящая к наиболее хорошей оценке с помощью целевой функции на этом уровне, будет представлять лучший исходный материал для формирования оптимальных решений на следующем уровне организации.

Таким образом, рекомбинируя участки хромосомы, можно получать блоки, образующие структуры, которые соответствуют более хорошим решениям. То есть, начиная с первого уровня организации хромосом, находится элемент популяции, оптимальный для этого уровня. Он будет использоваться как строительный блок генотипа при формировании оптимальных решений второго уровня, заполняя неспецифицированные гены, и т.д. Таким образом, алгоритм поднимается по лестнице промежуточных локально оптимальных решений, одновременно улучшая искомое глобальное решение.

Цикл работы мобильного генетического алгоритма имеет три этапа: инициализация, предварительная фаза и фаза процессинга,

которые могут быть выполнены на любом уровне организации, гарантируя тем самым, что строительные блоки (участки хромосомы) следующего уровня будут достаточно конкурентоспособны, чтобы обеспечить эволюцию.

Решение вопроса о времени остановки алгоритма не является тривиальным. Генетический алгоритм может использоваться на каждом следующем уровне, пока не будет использовано некоторое фиксированное количество памяти или времени, или, если в течение некоторого заданного количества поколений не будет происходить какого-либо заметного улучшения.

Инициализация выполняется посредством создания популяции, содержащей по одной копии всех подстрок длиной k, что гарантирует наличие в популяции всех необходимых строительных блоков генотипа. Когда выполняется обработка исходной популяции, то за счет применения новых генетических операторов формируются оптимальные структуры хромосом. Для выбора наиболее подходящего блока необходимо оценить каждый член популяции. Размер популяции при такой схеме инициализации определяется как:

$$n = 2^k \left( \frac{L}{k} \right), \qquad (3.51)$$

поскольку размерность равна L, то существует $k^L$ комбинаций генов размера k, и для каждой комбинации существует $2^k$ различных бинарных комбинаций аллелей.

Предварительная фаза предназначена для увеличения пропорции оптимальных элементов в популяции. Выполняется только отбор, другие генетические операторы (разрезание, сплайсинг, мутация) не используются. Необходимость оценивать каждый элемент исходной популяции, конечно, не является оптимальной вычислительной процедурой. Однако если бы это требовалось при формировании каждого нового поколения популяции, то данная процедура стала бы практически очень дорогостоящей и малопривлекательной в вычислительном отношении. Оценка произ-

водится только один раз во время предварительной фазы, так как вследствие неизменяемости хромосом, значение их оптимальности сохраняется. Затем происходит отбор наиболее хороших строк, чтобы увеличить их процент в популяции. Одновременно через определенные интервалы времени количество элементов популяции сокращается, чтобы достичь уровня, который будет сохраняться постоянным на следующей стадии эволюции.

Фаза процессинга. После обогащения популяции конкурентоспособными элементами на предварительном этапе, производится обработка популяции, которая напоминает стандартный генетический алгоритм. В течение этой фазы отбор используется совместно с кроссинговером. Возможна и мутация строк.

Для рекомбинации строк переменной длины применяются два новых оператора CUT и SPLICE вместо обычного кроссинговера с фиксированным положением одной или двух точек рекомбинации (рис. 3.16).

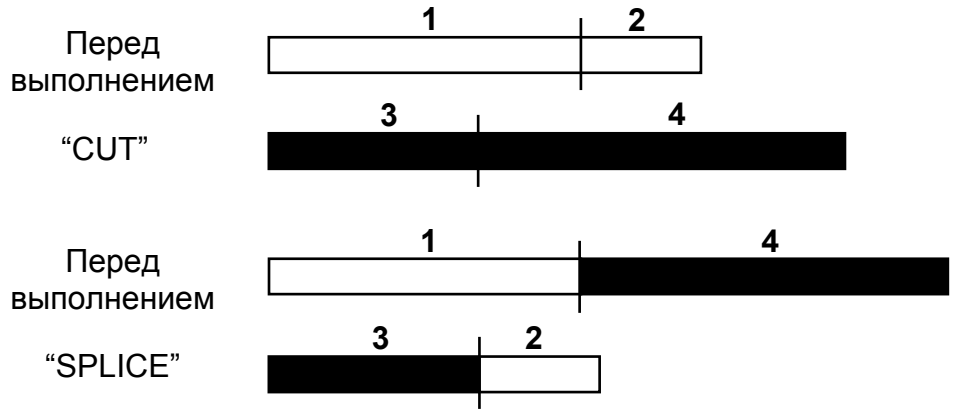


Рис. 3.16. *Функционирование операторов CUT и SPLICE*

Оператор CUT применяется к строке с вероятностью

$$p_c = (L - 1)p_k, \qquad (13.52)$$

где L – длина строки, $p_k$ – некоторая константа, равная, например, 0,1.

Оператор SPLICE объединяет две строки вместе с фиксированной вероятностью $p_s$.

Операторы CUT и SPLICE имеют два предельных типа пове-

дения. Первоначально, когда строки короткие, преобладает сцепление строк. Позднее, когда строки становятся достаточно длинными, преобладает деление строк.

Для анализа работы операторов CUT и SPLICE необходимо скорректировать понятие шаблона, так как в мобильном алгоритме гены могут находиться в произвольном месте хромосомы, и это не влияет на их интерпретацию и эффективность. Поэтому, в данном случае, шаблон – это кластер определенного подмножества генов, содержащих определенные аллели с определяющей длиной менее некоторого специфицированного значения.

Вероятность выживания шаблона H после применения оператора CUT, $P_{surv}$ можно охарактеризовать следующим неравенством:

$$P_{surv} \geq 1 - P_c \frac{\delta(H)}{\lambda - 1}, \qquad (3.53)$$

где $\sigma$ – определяющая длина шаблона; $\lambda$ – длина строки, содержащей шаблон; $p_c$ – вероятность операции CUT, определенная выше.

Подставляя выражение для вероятности CUT, получаем

$$P_{surv} \geq 1 - p_k \sigma(H). \qquad (13.54)$$

Вероятность выживания шаблона после операции SPLICE равна единице, так как она не вызывает эффектов разрушения.

Применяемые вместе, операторы CUT и SPLICE производят эффект подобный кроссинговеру стандартного генетического алгоритма. Однако физического выживания шаблона в мобильном алгоритме недостаточно, необходимо еще, чтобы он был экспрессирован. Только после этого он начинает играть активную роль в поиске оптимума. Рассмотрим вероятность $P_\xi$ продолжения экспрессии шаблона, активного в данный момент. Пусть N – событие, состоящее в том, что шаблон, экспрессированный на предыдущем шаге, не экспрессируется после последнего выполнения операторов CUT и SPLICE. Тогда

$$P_\xi = 1 - p_s P(N). \qquad (3.55)$$

Экспрессия может быть обусловлена следующими двумя событиями: событие F – рассматриваемый сегмент строки находится вначале сцепленной пары строк, событие B – сегмент находится в конце сцепленных строк. Тогда можно записать

$$P(N) = P(N|F)\,P(F) + P(N|B)\,P(B) \qquad (3.56)$$

Если подстрока в настоящий момент экспрессируется, и она помещается в начало сцепленной пары, тогда шансы, что она не будет экспрессирована, нулевые, P(N|F) = 0. С другой стороны, если эта же подстрока помещается в конец сцепленной пары, то вероятность ее экспрессии, P(N|B), будет зависеть от количества и типа генов, расположенных перед ней. Оценка этой вероятности для общего случая очень трудна. Поэтому, накладывая некоторые разумные предположения, можно получить приближенную оценку этой величины. Если гены распределены равномерно случайно по популяции и появление любого, отличного от анализируемого, гена в начале сцепленной пары блокирует экспрессию этого гена. При этих условиях

$$P(N \mid B) \le 1 - \left(1 - \frac{k}{l}\right)^{\lambda^*}, \qquad (3.57)$$

где $\lambda^*$ – максимальная длина строки в популяции.

Применяя теорему о биномиальном распределении и отбрасывая члены высоких порядков, получаем:

$$P(N \mid B) \le \frac{k\lambda^*}{L}. \qquad (3.58)$$

Учитывая, что оператор SPLICE помещает подстроку в начало или конец новой строки с равной вероятностью, имеем:

$$P_\xi \ge 1 - p_s \frac{k\lambda^*}{2L}. \qquad (3.59)$$

Видно, что вероятность экспрессии очень велика при малых длинах строк и быстро уменьшается при росте длины строки.

Вычисляя общую вероятность выживания и экспрессии шаблона, $P_{se}$, как произведение отдельных вероятностей, получаем

$$P_{se} \geq (1 - p_k \delta(H)) \left( 1 - \frac{p_s k \lambda^*}{2L} \right). \qquad (3.60)$$

Это уравнение можно свести к более простой форме после отбрасывания членов высоких порядков

$$P_{se} \geq (1 - p_k \delta(H) - p_s \frac{k \lambda^*}{2L}). \qquad (3.61)$$

Таким образом, применяемые вместе, операторы CUT и SPLICE производят примерно тот же результат, что стандартный кроссинговер с одной точкой рекомбинации, при малых длинах строк эффекты разрушения незначительны, при увеличении длины вероятность невыполнения экспрессии значительно увеличивается.

* * *

В заключение еще раз отметим, что в отличие от стандартного генетического, мобильный алгоритм не накладывает ограничений на позиционирование генов внутри хромосом, допуская их различные перестановки внутри строк, в результате чего не требуется предварительная информация о наиболее оптимальном расположении бит внутри строки, что очень важно в приложениях.

В настоящее время наблюдается столь бурное и всестороннее развитие оптимизационных моделей, что становится затруднительным дать исчерпывающую оценку степени их применимости для практического решения системных проблем. Можно отметить лишь следующее.

Как показывает исследовательская практика, пока ни одна из самых совершенных моделей оптимизации, взятая в отдельности, не позволяет разрешить сколько-нибудь значимую системную проблему. Те отдельные успехи, когда удалось свести реальную проблему к задаче оптимизации, решить ее и получить результаты, удовлетворяющие заказчика – уникальны и заслуживают всеобщего восхищения. В большинстве же случаев имеет место ситуация, когда желаемое выдается за действительное. Более того, многочисленные ограничения и допущения, возникающие в процессе разработки оптимизационных моделей, столь тщательно скрываются и маскируются, что у лица, принимающего решение на основе результатов такой «оптимизации», остается только один непоколебимый аргумент – для обоснования решения использовались компьютерные математические модели.

Тем не менее, оптимизационные модели занимали, занимают и будут занимать важное место как в теории, так и в практике системного моделирования, позволяя успешно решать множество частных задач, составляющих общую системную проблему, и, самое главное, помогая исследователям глубже проникнуть в существо изучаемого объекта.

# ГЛАВА 4. НЕЙРОСЕТЕВОЕ МОДЕЛИРОВАНИЕ

## 4.1. Идея

Нейросетевым моделированием называется способ имитации процессов функционирования реальных систем на основе искусственных нейронных сетей. Искусственные нейронные сети, они же коннекционистские (от англ. *connection* – связь) системы, представляют собой устройства, использующие огромное число взаимосвязанных элементарных условных рефлексов, названных по имени канадского физиолога Д. Хебба – «синапсами Хебба».

В настоящее время нейросетевое моделирование применяется для решения задач обработки изображений, управления роботами и непрерывными производствами, для понимания и синтеза речи, для диагностики заболеваний людей и обнаружения технических неполадок в приборах, для синтеза корпоративных информационных систем и имитации пищевых технологических процессов. Та часть работ, которая связана с разработкой устройств переработки информации на основе принципов работы нейронных систем, относится к области нейроинформатики.

Суть всех подходов нейроинформатики заключается в разработке методов синтеза нейронных сетей, имитирующих процессы функционирования различных систем и позволяющих решать те или иные задачи. Нейрон при этом выглядит как очень простое устройство: нечто вроде усилителя с большим числом входов и одним выходом. Различие между подходами и методами – в деталях представлений о работе нейрона, и, конечно, в представлениях о работе связей. Собственно, нейросети представляют собой ни что иное, как связевые системы. В отличие от обычных цифровых микропроцессорных систем, представляющих собой сложные комбинации процессоров и запоминающих блоков, в искусственных нейросетях память и операции сосредоточены в связях между процессорами-нейронами. Тем самым основная нагрузка на выполнение конкретных функций ложится не на отдельные процес-

соры, а на всю архитектуру системы, структура которой определяется межнейронными связями.

Таким образом, ядром нейросетевого моделирования выступает паллиативная идея о том, что нейроны можно моделировать довольно простыми автоматами, а вся сложность мозга, гибкость его функционирования и другие важнейшие качества определяются связями между нейронами. В этой идее ключевым в понимании философской сущности нейросетевого моделирования (впрочем, как и, рассмотренного ранее, логико-лингвистического) выступает слово «паллиативная» (от познелат. *pallio* – прикрываю), означающее в современном толковании меру, не обеспечивающую полного, коренного решения поставленной задачи. По-видимому, прав был Френсис Бэкон, когда утверждал, что «*человек, слуга и истолкователь Природы, ровно столько совершает и понимает, сколько он охватывает в порядке Природы; свыше этого он не знает и не понимает ничего*».

С центральной идеей нейросетевого моделирования, предельным выражением которой служит фраза: «связи – все, свойства элементов – ничто», тесно связаны следующие аксиомы:

1) несмотря на то, что элементы, из которых строится нейросетевая модельная среда, однородны и чрезвычайно просты, с их помощью можно имитировать процессы любой сложности;

2) из простых и ненадежных элементов можно построить вполне надежную систему, когда при разрушении случайно выбранной части система сохраняет свои полезные свойства;

3) предполагается, что нейросеть достаточно богата по своим возможностям и достаточно избыточна, чтобы компенсировать бедность выбора элементов, их ненадежность, возможные разрушения части связей.

Названные аксиомы, определяя принцип построения нейросетевых моделей, не указывают на то, как же их строить и научить решать реальные задачи. На первый взгляд кажется, что нейросети не допускают прямого программирования, то есть формирова-

ния связей по явным правилам. Это, однако, не совсем так. Существует большой класс задач (нейронные системы ассоциативной памяти, статистической обработки, фильтрации и др.), для которых связи формируются по явным формулам. Но еще большее (по объему существующих приложений) число задач требует неявного процесса. По аналогии с обучением животных или человека этот процесс называется обучением нейросетевой модели.

Методы обучения нейросетей достаточно просты и имеют несложную практическую реализацию. Обучение обычно строится так: существует задачник – набор примеров с заданными ответами. Эти примеры предъявляются системе. Нейроны получают по входным связям сигналы – «условия примера», преобразуют их, несколько раз обмениваются преобразованными сигналами и, наконец, выдают ответ – также набор сигналов. Отклонение от правильного ответа штрафуется. Обучение состоит в минимизации штрафа как (неявной) функции связей.

Обучение приводит к тому, что структура связей становится «непонятной» – не существует иного способа ее прочитать, кроме как запустить функционирование сети. Становится сложно ответить на вопрос: «Как нейронная сеть получает результат?» – то есть построить понятную человеку логическую конструкцию, воспроизводящую действия сети. Это явление можно назвать «логической непрозрачностью» нейронных сетей. В работе с логически непрозрачными нейронными сетями оказываются полезными представления, разработанные в психологии и педагогике [Эсаулов, 1972], и обращение с обучаемой сетью как с объектом дрессировки или натаскивания. С другой стороны, при использовании нейросетевых моделей возникает потребность прочитать и логически интерпретировать навыки, выработанные сетью. Методы получения неявными приемами логически прозрачных нейронных называются контрастированием. Следует отметить, что за логическую прозрачность нейросетевых моделей приходится платить снижением избыточности, так как при контрастировании удаля-

ются все связи, кроме важнейших, без которых задача не может быть решена.

Для решения некоторых типов задач уже существуют оптимальные конфигурации нейросетевых моделей. В общем же случае подбор оптимальной архитектуры сети является трудоемкой слабоформализуемой задачей, при решении которой обычно руководствуются следующими эвристическими правилами, заимствованными из теории проектирования сложных систем:

- возможности сети возрастают с увеличением числа ее ячеек, то есть, чем больше ячеек в нейросетевой модели, тем адекватнее имитируется изучаемый процесс;
- плотность связей между нейронами оказывает положительное влияние на характер процесса обучения, то есть, чем теснее связаны нейроны в сети, тем быстрее и полнее идет процесс ее обучения;
- сложность алгоритмов функционирования сети способствует усилению ее мощи, то есть, чем разветвленнее сеть, тем выше ее имитационные возможности;
- введение обратных связей повышает адаптивные способности нейросетевой модели, но одновременно может привести к снижению ее динамической устойчивости, то есть, возникают ситуации, когда уже обученная нейронная сеть начинает вести себя непредсказуемым образом.

Итак, становится очевидным наличие трех источников идеологии нейросетевого моделирования – это имеющиеся на сегодняшний день представления о строении и функционировании мозга, современное понимание процессов обучения и общие положения проектирования сложных систем. Далее будут рассмотрены некоторые вопросы построения нейросетевых моделей и их применения для решения отдельных задач.[*)]

[*)] При написании разделов 4.2 и 4.3 использованы материалы лекций А.Н.Горбань (ВЦ СО РАН г. Красноярск) на Всероссийской школе «Нейроинформатика-96».

## 4.2. Элементы и архитектура нейросетевых моделей

Для описания нейросетевых моделей используется специальная «схемотехника», в которой элементарные устройства – сумматоры, синапсы, нейроны и т.п. объединяются в сети, предназначенные для решения задач. Любопытен статус этой «схемотехники» – ни в аппаратной реализации нейронных сетей, ни в профессиональном программном обеспечении все эти элементы не обязательно реализуются как отдельные части или блоки. Используемая в нейросетевом моделировании, «схемотехника» представляет собой особый язык для представления нейронных сетей и их обсуждения. При программной и аппаратной реализации, описания, выполненные на этом языке, переводятся на языки другого уровня, более пригодные для программно-компьютерного исполнения.

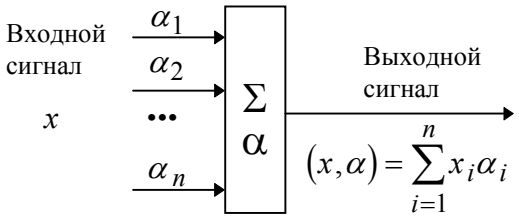


Рис. 4.1. *Адаптивный сумматор*

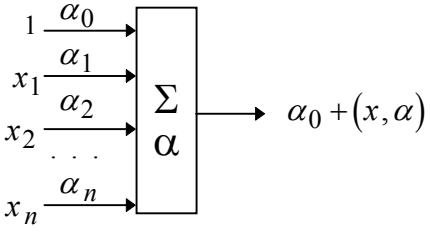


Рис. 4.2. *Неоднородный адаптивный сумматор*

Наиболее важный элемент любой нейросетевой модели – это адаптивный сумматор (рис. 4.1), который вычисляет скалярное произведение вектора входного сигнала x на вектор параметров α. Адаптивным он называется из-за наличия вектора настраиваемых параметров α. Для многих задач полезно иметь неоднородную линейную функцию выходных сигналов. Ее вычисление также можно представить с помощью неоднородного адаптивного сумматора, имеющего n + 1 вход и получающего на 0-й вход постоянный единичный сигнал (рис. 4.2).

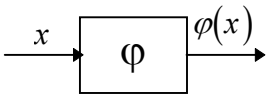


Рис. 4.3. *Нелинейный преобразователь сигнала*

Следующий типовой элемент нейросетевой модели называется нелинейным преобразователем сигнала (рис.

4.3). Он получает скалярный входной сигнал x и переводит его в φ(x), где φ – функция активации нейрона.

Точка ветвления (рис. 4.4) служит для рассылки одного сигнала по нескольким адресам. Она получает скалярный входной сигнал x и передает его всем своим выходам.

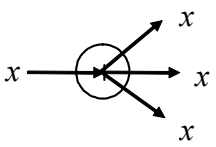


Рис. 4.4. *Точка ветвления*

Стандартный формальный нейрон составляется из входного сумматора, нелинейного преобразователя и точки ветвления на выходе (рис. 4.5). Линейная связь (синапс) отдельно от сумматоров не встречается, однако для некоторых рассуждений бывает удобно выделить этот элемент (рис. 4.6). Он умножает входной сигнал x на «вес синапса» α.

Также бывает полезно «присоединить» связи не к входному сумматору, а к точке ветвления. В результате получаем элемент, двойственный адаптивному сумматору и называемый «выходная звезда». Его выходные связи производят умножение сигнала на свои веса.

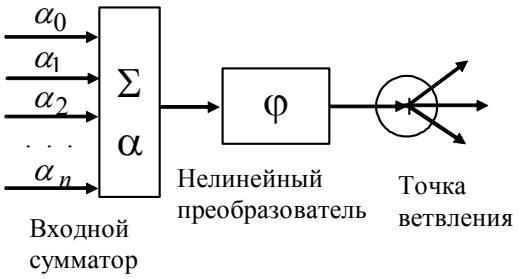


Рис. 4.5. *Стандартный формальный нейрон*

Перейдем теперь к вопросу составления нейросетевых моделей из указанных элементов. Вообще говоря, эти элементы можно соединять произвольным образом, лишь бы входы получали какие-нибудь сигналы. Но такой произвол выливается в трудности обучения модели, поэтому на практике используют несколько стандартных архитектур, из которых строят большинство используемых нейросетевых моделей.

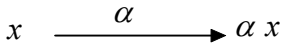


Рис. 4.6. *Синапс*

Можно выделить две базовых архитектуры нейросетевых моделей – слоистые и полносвязные. В слоистых моделях нейроны расположены в нескольких слоях (рис. 4.7). Нейроны первого слоя получают входные сигналы, преобразуют их и через точки ветвления передают нейронам второго слоя. Далее срабатывает второй

слой и так до k-го слоя, который выдает выходные сигналы для интерпретатора и пользователя. Если не оговорено противное, то каждый выходной сигнал i-го слоя подается на вход всех нейронов (i + 1)-го. Число нейронов в каждом слое может быть любым и заранее никак не связано с количеством нейронов в других слоях. Стандартный способ подачи входных сигналов: все нейроны первого слоя получают каждый входной сигнал. Особое распространение получили трехслойные сети, в которых каждый слой имеет свое наименование: первый – входной, второй – скрытый, третий – выходной.

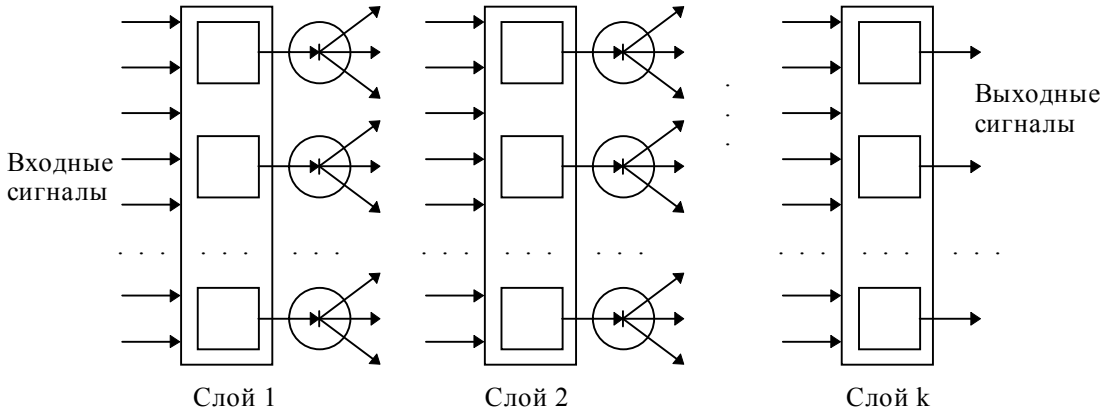


Рис. 4.7. *Слоистая нейросетевая модель*

В полносвязных моделях каждый нейрон передает свой выходной сигнал остальным нейронам, включая самого себя (обратная связь). Выходными сигналами сети могут быть все или некоторые выходные сигналы нейронов после нескольких тактов функционирования сети. Все входные сигналы подаются всем нейронам.

Элементы слоистых и полносвязных сетей могут выбираться по-разному, однако существует стандартное правило: на входе должен находиться нейрон с адаптивным неоднородным линейным сумматором на входе. Для полносвязной сети входной сумматор нейрона фактически распадается на два: первый вычисляет линейную функцию от входных сигналов сети, второй – линейную функцию от выходных сигналов других нейронов, полученных на предыдущем шаге.

*Функция активации нейронов (характеристическая функция)* φ, преобразующая выходной сигнал сумматора, может быть одной и той же для всех нейронов сети. В этом случае нейросетевую модель называют *однородной (гомогенной)*. Если же φ зависит еще от одного или нескольких параметров, значения которых меняются от нейрона к нейрону, то модель называют *неоднородной (гетерогенной)*.

Функция активации нейрона может иметь различный вид и оказывает существенное влияние на характеристики нейросетевой модели. Одной их наиболее распространенной функцией активации является логистическая функция или сигмоид

$$\varphi(t) = \frac{1}{1 + e^{-\beta t}}, \qquad (4.1)$$

где β - параметр, подбираемый пользователем и влияющий на пологость функции.

Составление модели из нейронов стандартного вида (рис. 4.5) не является обязательным. Слоистая или полносвязная архитектуры не налагают существенных ограничений на участвующие в них элементы. Единственное жесткое требование, предъявляемое архитектурой к элементам модели, это соответствие размерности вектора входных сигналов элемента (она определяется архитектурой) числу его входов.

Если полносвязная модель функционирует до получения ответа заданное число тактов k, то ее можно представить как частный случай k-слойной модели, все слои которой одинаковы, и каждый из них соответствует такту функционирования полносвязной модели. Существенное различие между полносвязной и слоистой моделями возникает тогда, когда число тактов функционирования заранее не ограничено – слоистая модель так работать не может.

## 4.3. Некоторые задачи, решаемые с помощью нейросетевых моделей

В настоящее время нейросетевые модели используются в самых различных областях для решения весьма обширного круга практических проблем, которые условно можно разделить на следующие группы: вычисления и аппроксимация функций; классификация и распознавание образов; прогнозирование; идентификация и оценивание; ассоциативное управление. Рассмотрим некоторые простейшие задачи, которые можно успешно решать с помощью нейросетевых моделей, имея в виду не более чем демонстрацию возможностей нейросетевого подхода. Даже простейшие нейросетевые модели, состоящие из одного адаптивного сумматора, находят широкое применение для решения следующих задач.

### 4.3.1. Построение линейной регрессии

Суть задачи состоит в поиске наилучшего линейного приближения функции, заданной конечным набором значений: дана выборка значений вектора аргументов $x^1$, ..., $x^m$, заданы значения функции F в этих точках: $F(x^i) = f_i$, требуется найти линейную (неоднородную) функцию $\varphi(x) = (\alpha, x) + \alpha_0$, ближайшую к F. Чтобы однозначно поставить задачу, необходимо доопределить, что значит «ближайшую». Для этого воспользуемся методом наименьших квадратов, согласно которому $\varphi$ ищется из условия

$$\sum_{i=1}^{m} [F(x^i) - \varphi(x^i)]^2 \to \min . \qquad (4.2)$$

Необходимо подчеркнуть, что метод наименьших квадратов не является ни единственным, ни наилучшим во всех отношениях способом доопределения задачи регрессии. Его главное достоинство - квадратичность критерия и линейность получаемых уравнений относительно коэффициентов $\varphi$.

Явные формулы линейной регрессии легко получить, минимизируя квадратичный критерий качества регрессии. Обозначим

$$\Delta_i = [F(x^i) - \varphi(x^i)]; H = \sum_{i=1}^{m} \Delta_i^2 = \sum_{i=1}^{m} [F(x^i) - \varphi(x^i)]^2 = \\ = \sum_{i=1}^{m} [f_i - (\alpha, x^i) - \alpha_0]^2. \quad (4.3)$$

Найдем производные минимизируемой функции H по настраиваемым параметрам:

$$\frac{\partial H}{\partial \alpha_j} = \sum_{i=1}^{m} \Delta_i x_j^i, (j = 1,..., n); \frac{\partial H}{\partial \alpha_0} = \sum_{i=1}^{m} \Delta_i , \quad (4.4)$$

где $x_j^i$ – j-я координата вектора $x^i$.

Приравнивая частные производные H нулю, получаем уравнения, из которых легко найти все $\alpha_j$ (j = 0,..., n). Решение удобно записать в общем виде, если для всех i = 1,..., m обозначить $x_0^i \equiv 1$ и рассматривать (n + 1)-мерные векторы данных $x^i$ и коэффициентов $\alpha$. Тогда

$$\Delta_i = f_i - (\alpha, x^i), \partial H / \partial \alpha_j = \sum_{i=1}^{m} \Delta_i x_j^i \ \ (j = 0,1,..., n). \quad (4.5)$$

Обозначим символом p(n 0+ 1)-мерный вектор с координатами $p_j = \frac{1}{m} \sum_{i=1}^{m} f_i x_j^i$, а символом Q – матрицу размером (n + 1) × (n + 1) с элементами $q_{jk} = \frac{1}{m} \sum_{i=1}^{m} x_j^i x_k^i$ .

В новых обозначениях решение задачи линейной регрессии будет иметь вид:

$$\varphi(x) = (\alpha, x), \alpha = Q^{-1}p. \quad (4.6)$$

Приведем ее решение в традиционных обозначениях математической статистики. Обозначим $M_o$ среднее значение j-й координаты векторов исходной выборки:

$$M_j = \frac{1}{m} \sum_{i=1}^{m} x_j^i . \quad (4.7)$$

Пусть M – вектор с координатами $M_o$. Введем также обозначение $s_j$ для выборочного среднеквадратичного отклонения:

$$s_j = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(x_j^i - M_j)^2} \qquad (4.8)$$

Величины $s_j$ задают естественный масштаб для измерения j-х координат векторов x. Кроме того, нам потребуются величина $s_f$ и коэффициенты корреляции f с j-ми координатами векторов x - $r_{fj}$:

$$s_f = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(f_i - M_f)^2}, M_f = \frac{1}{m}\sum_{i=1}^{m} f_i, r_{fj} = \frac{\frac{1}{m}\sum_{i=1}^{m}(f_i - M_f)(x_j^i - M_j)}{s_f s_j}. \qquad (4.9)$$

Вернемся к n-мерным векторам данных и коэффициентов. Представим, что векторы сигналов проходят предобработку – центрирование и нормировку и далее мы имеем дело с векторами $y^i$:

$$y_j^i = \frac{x_j^i - M_j}{s_j}. \qquad (4.10)$$

Это, в частности, означает, что все рассматриваемые координаты вектора x имеют ненулевую дисперсию, то есть постоянные координаты исключаются из рассмотрения, поскольку они не несут полезной информации. Уравнения регрессии будем искать в форме: $\varphi(y) = (\beta, y) + \beta_0$. Получим:

$$\beta_0 = M_f, \beta = s_f R^{-1} R_f, \qquad (4.11)$$

где $R_f$ – вектор коэффициентов корреляции f с j-ми координатами векторов x, имеющий координаты $r_{fj}$; R – матрица коэффициентов корреляции между координатами вектора данных:

$$r_{kj} = \frac{\frac{1}{m}\sum_{i=1}^{m}(x_k^i - M_k)(x_j^i - M_j)}{s_k s_j} = \frac{1}{m}\sum_{i=1}^{m} y_k^i y_j^i. \qquad (4.12)$$

В задачах обработки данных почти всегда возникает вопрос о последовательном уточнении результатов по мере поступления

новых данных. Существует как минимум два подхода к ответу на этот вопрос для задачи линейной регрессии. Первый подход состоит в том, что изменения в коэффициентах регрессии при поступлении новых данных рассматриваются как малые и при их вычислении ограничиваются первыми порядками. При втором подходе для каждого нового вектора данных делается шаг изменений коэффициентов, уменьшающий ошибку регрессии $\Delta^2$ на вновь поступившем векторе данных. При этом «предыдущий опыт» фиксируется только в текущих коэффициентах регрессии.

В рамках первого подхода рассмотрим, как изменится $\alpha$ при добавлении новых данных. В первом порядке найдем изменение вектора коэффициента $\alpha$ при изменении вектора p и матрицы Q:

$$\begin{aligned}
&\alpha + \Delta\alpha = (Q + \Delta Q)^{-1}(p + \Delta p);\\
&(Q + \Delta Q)^{-1} = (Q(1 + Q^{-1}\Delta Q))^{-1} = Q^{-1} - Q^{-1}\Delta Q Q^{-1} + o(\Delta Q); \quad (4.13)\\
&\Delta\alpha \cong Q^{-1}(\Delta p - \Delta Q\alpha).
\end{aligned}$$

Пусть на выборке $\{x^i\}_{i=1}^m$ вычислены p, Q и $Q^{-1}$. При получении нового вектора данных $x^{m+1}$ и соответствующего ему значения $F(x^{m+1}) = f_{m+1}$ имеем:

$$\begin{aligned}
&\Delta p = \tfrac{1}{m+1}(f_{m+1}x^{m+1} - p);\\
&\Delta q_{jk} = \tfrac{1}{m+1}(x_j^{m+1}x_k^{m+1} - q_{jk});\\
&\Delta Q = \tfrac{1}{m+1}[(x^{m+1}) \otimes (x^{m+1})^T]; \qquad (4.14)\\
&\Delta(Q^{-1}) = \tfrac{1}{m+1}[Q^{-1} - (Q^{-1}x^{m+1}) \otimes (Q^{-1}x^{m+1})^T];\\
&\Delta\alpha = \tfrac{1}{m+1}\Delta_{m+1}^0 Q^{-1}x^{m+1},
\end{aligned}$$

где $\Delta_{m+1}^0 = f_{m+1} - (\alpha, x^{m+1})$ – ошибка на векторе данных $x^{m+1}$ регрессионной зависимости, полученной на основании выборки $\{x^i\}_{i=1}^m$; $\otimes$ – произведение вектора-столбца на вектор-строку (тензорное произведение).

Пересчитывая по приведенным формулам p, Q, $Q^{-1}$ и $\alpha$ после каждого получения данных, получаем процесс, в котором после-

довательно уточняются уравнения линейной регрессии. Требуемый объем памяти, и количество операций имеют порядок $n^2$ - из-за необходимости накапливать и модифицировать матрицу $Q^{-1}$. Конечно, это меньше, чем потребуется на обычное обращение матрицы Q на каждом шаге, однако следующий простой алгоритм еще экономнее. Его особенность заключается в том, что он не обращается к матрицам Q, $Q^{-1}$ и основан на уменьшении на каждом шаге величины $(\Delta^0_{m+1})^2 = [f_{m+1} - (\alpha, x^{m+1})]^2$ – квадрата ошибки на векторе данных $x^{m+1}$ регрессионной зависимости, полученной на основании выборки $\{x^i\}_{i=1}^m$.

Рассмотрим вопрос обучения адаптивного сумматора. Для этого обратимся к формуле (12.6) и будем анализировать (n + 1)-мерные векторы данных и коэффициентов.

Обозначим $x = x^{m+1}; \Delta = \Delta^0_{m+1} = f_{m+1} - (\alpha, x^{m+1})$. Тогда

$$\mathrm{grad}_\alpha \Delta = -\Delta \times x. \qquad (4.15)$$

Эта выражение носит название формулы Уидроу, а процедура ее использования для обучения адаптивного сумматора получила название метода наискорейшего спуска. Суть этого метода состоит в изменении вектора коэффициентов $\alpha$ в направлении антиградиента $\Delta^2$: на каждом шаге к $\alpha$ добавляется $h \times \Delta \times x$, где h – величина шага.

Если при каждом поступлении нового вектора данных x изменять $\alpha$ указанным образом, то получим последовательную процедуру построения линейной аппроксимации функции F(x). Такой алгоритм обучения легко реализуется аппаратными средствами (изменение веса связи $\alpha_j$ есть произведение прошедшего по ней сигнала $x_j$ на ошибку $\Delta$ и на величину шага). Возникает, однако, проблема сходимости (если h слишком мало, то сходимость будет медленной, если же слишком велико, то произойдет потеря устойчивости и сходимости не будет вовсе), решение которой достигается экспериментальными методами [Уидроу, Стирнз, 1989].

### 4.3.2. Линейное разделение двух классов

Задача формулируется так: имеется два набора векторов $x^1$ , ..., $x^m$ и $y^1$,..., $y^m$ . Заранее известно, что $x^i$ относится к первому классу, а $y^i$ - ко второму. Требуется построить решающее правило, то есть определить такую функцию f(x), что при f(x) > 0 вектор x относится к первому классу, а при f(x) < 0 – ко второму.

Эта задача возникает во многих случаях: при диагностике болезней и определении неисправностей машин по косвенным признакам, при распознавании изображений и сигналов и т.п.

Строго говоря, классифицируются не векторы свойств, а объекты, которые обладают этими свойствами. Это замечание становится важным в тех случаях, когда возникают затруднения с построением решающего правила, например, тогда, когда встречаются принадлежащие к разным классам объекты, имеющие одинаковые признаки. В этих случаях возможно несколько решений:

1) искать дополнительные признаки, позволяющие разделить классы;

2) признать неизбежность ошибок и ввести штрафы ($c_{12}$ - штраф за то, что объект первого класса отнесен ко второму, $c_{21}$ - за то, что объект второго класса отнесен к первому) и строить разделяющее правило так, чтобы минимизировать математическое ожидание штрафа;

3) перейти к нечеткому разделению классов – строить так называемые функции принадлежности $f_1(x)$ и $f_2(x)$, которые оценивают степень уверенности при отнесении объекта к i-му классу (i = 1, 2).

Линейное разделение классов состоит в построении линейного решающего правила, то есть такого вектора $\alpha$ и числа $\alpha_0$ (называемого порогом), что при $(x, \alpha) > \alpha_0$ x относится к первому классу, а при $(x, \alpha) < \alpha_0$ – ко второму.

Поиск такого решающего правила можно рассматривать как разделение классов в проекции на прямую. Вектор $\alpha$ задает пря-

мую, на которую ортогонально проектируются все точки, а число $\alpha_0$ – точку на этой прямой, отделяющую первый класс от второго.

Простейший выбор состоит в проектировании на прямую, соединяющую центры масс выборок. Центр масс вычисляется в предположении, что массы всех точек одинаковы и равны 1. Это соответствует заданию $\alpha$ в виде

$$\alpha = \frac{1}{m}(y^1 + y^2 + ... + y^m) - \frac{1}{n}(x^1 + x^2 + ... + x^n). \qquad (4.16)$$

Во многих случаях удобнее иметь дело с векторами единичной длины. Нормируя $\alpha$, получаем:

$$\alpha = \frac{1}{к}[\frac{1}{m}(y^1 + y^2 + ... + y^m) - \frac{1}{n}(x^1 + x^2 + ... + x^n)] \qquad (4.17)$$

где $к = \|\frac{1}{m}(y^1 + y^2 + ... + y^m) - \frac{1}{n}(x^1 + x^2 + ... + x^n)\|$.

Выбор $\alpha_0$ может производиться из различных соображений. Простейший вариант – посередине между центрами масс выборок. Более тонкие способы построения границы раздела классов $\alpha_0$ учитывают различные вероятности появления объектов разных классов, и оценки плотности распределения точек классов на прямой. Чем меньше вероятность появления данного класса, тем более граница раздела приближается к центру тяжести соответствующей выборки.

Можно для каждого класса построить приближенную плотность вероятностей распределения проекций его точек на прямую (это намного проще, чем для многомерного распределения) и выбирать $\alpha_0$, минимизируя вероятность ошибки. Пусть решающее правило имеет следующий вид: при $(x, \alpha) > \alpha_0$ x относится к первому классу, а при $(x, \alpha) < \alpha_0$ - ко второму. В таком случае вероятность ошибки будет равна

$$P = p_1 \int_{-\infty}^{\alpha_0} \rho_1(\chi) d\chi + p_2 \int_{\alpha_0}^{\infty} \rho_2(\chi) d\chi \qquad (4.18)$$

где $p_1$, $p_2$ – априорные вероятности принадлежности объекта соответствующему классу, $\rho_1(\chi)$, $\rho_2(\chi)$ - плотности вероятности для распределения проекций $\chi$ точек x в каждом классе.

Приравняв нулю производную вероятности ошибки по $\alpha_0$, получим: число $\alpha_0$, доставляющее минимум вероятности ошибки, является корнем уравнения:

$$p_1\rho_1(\chi) = p_2\rho_2(\chi), \qquad (4.19)$$

либо (если у этого уравнения нет решений) оптимальным является правило, относящее все объекты к одному из классов.

Если принять гипотезу о нормальности распределений:

$$\rho(y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(y-a)^2/2\sigma^2},$$

то для определения $\alpha_0$ получим:

$$\frac{p_1}{\sqrt{2\pi}\sigma_1} e^{-(y-a_1)^2/2\sigma_1^2} = \frac{p_2}{\sqrt{2\pi}\sigma_2} e^{-(y-a_2)^2/2\sigma_2^2}, \qquad (4.20)$$

$$\ln(p_1/p_2) - \ln(\sigma_1/\sigma_2) - (y-a_1)^2/2\sigma_1^2 + (y-a_2)^2/2\sigma_2^2 = 0. \qquad (4.21)$$

Если уравнение (4.21) имеет два корня $y = \alpha_1, \alpha_2$, $(\alpha_1 < \alpha_2)$, то наилучшим решающим правилом будет: при $\alpha_1 < (x, \alpha) < \alpha_2$ объект принадлежит одному классу, а при $\alpha_1 > (x, \alpha)$ или $(x, \alpha) > \alpha_2$ – другому (к какому именно, определяется тем, которое из произведений $p_i\rho_i(\chi)$ больше). Случай единственного корня представляет интерес только тогда, когда $\sigma_1 = \sigma_2$. При этом уравнение (4.21) становится линейным, и мы приходим к исходному варианту - единственной разделяющей точке $\alpha_0$.

Если рассматривать задачу об оптимального разделения многомерных нормальных распределений, то наилучшей разделяющей поверхностью является квадрика (на прямой – две точки). Предполагая, что ковариационные матрицы классов совпадают (в одномерном случае это предположение о том, что $\sigma_1 = \sigma_2$), получаем линейную разделяющую поверхность, которая ортогональна прямой, соединяющей центры выборок не в обычном скалярном

произведении, а в специальном: $\langle x, y\rangle = \left(x, \Sigma^{-1} y\right)$, где $\Sigma$ – общая ковариационная матрица классов [Дуда, Харт, 1976].

Усовершенствовать разделяющее правило можно путем использования не просто вероятности ошибки, а среднего риска. В этом случае каждой ошибке приписывается «цена» $c_i$ и минимизируется сумма $c_1 p_1 \rho_1(\chi) + c_2 p_2 \rho_2(\chi)$. Ответ получается практически тем же ($p_i$ меняются на $c_i p_i$), но такая добавка важна для многих приложений.

Требование безошибочности разделяющего правила на обучающей выборке принципиально отличается от обсуждавшихся критериев оптимальности. На основе этого требования строится персептрон Ф. Розенблатта [Розенблатт, 1965] .

Возьмем за основу при построении гиперплоскости, разделяющей классы, отсутствие ошибок на обучающей выборке. Чтобы удовлетворить этому условию, придется решать систему линейных неравенств:

$$(x^i, \alpha) > \alpha_0 \ (i = 1,..., n)$$
$$(y^j, \alpha) < \alpha_0 \ (j = 1,..., m). \qquad (4.22)$$

Здесь $x^i$ (i = 1,.., n) – векторы из обучающей выборки, относящиеся к первому классу, а $y^j$ (j = 1,..., n) – ко второму.

Переформулируем задачу. Увеличим размерности всех векторов на единицу, добавив еще одну координату – $\alpha_0$ к $\alpha$, $x_0 = 1$ – ко всем x и $y_0 = 1$ – ко всем y . Сохраним для новых векторов прежние обозначения и положим $z^i = x^i$ (i = 1,..., n), $z^j = -y^j$ (j = 1,..., m).

Тогда получим систему неравенств $(z^i, \alpha) > 0$ (i = 1,..., n + m), которую будем решать относительно $\alpha$. Если множество решений не пусто, то любой его элемент $\alpha$ порождает решающее правило, безошибочное на обучающей выборке.

Итерационный алгоритм решения этой системы основан на том, что для любого вектора x его скалярный квадрат (x,x) больше нуля. Пусть $\alpha$ – некоторый вектор, претендующий на роль реше-

ния неравенств $(z^i,\alpha) > 0$ (i = 1,..., n + m), однако часть из них не выполняется. Прибавим те $z^i$, для которых неравенства имеют неверный знак, к вектору $\alpha$ и вновь проверим все неравенства $(z^i,\alpha) > 0$ и т.д. Если они совместны, то процесс сходится за конечное число шагов. Более того, добавление $z^i$ к $\alpha$ можно производить сразу после того, как ошибка $[(z^i, \alpha) < 0]$ обнаружена, не дожидаясь проверки всех неравенств – и этот вариант алгоритма тоже сходится [Минский, Пайперт, 1971].

Перейдем от одноэлементных систем к нейронным сетям.

### 4.3.3 Вычисление непрерывных функций многих переменных и аппроксимация непрерывных автоматов

Доказательство того, что с помощью нейронных сетей можно с любой наперед заданной точностью вычислять произвольную непрерывную функцию $f(x_1,..., x_n)$, основывается на теореме А.Н. Колмогорова [Колмогоров, 1957]: каждая непрерывная функция n переменных, заданная на единичном кубе n-мерного пространства, представима в виде

$$f(x_1, x_2,..., x_n) = \sum_{q=1}^{2n+1} h_q \left[ \sum_{p=1}^{n} \varphi_q^p(x_p) \right], \qquad (4.23)$$

где функции $h_q(u)$ непрерывны, а функции $\varphi_q^p(x_p)$, кроме того, еще и стандартны, то есть не зависят от выбора функции f.

Из этой теоремы в частности следует, что можно получить любую непрерывную функцию n переменных с помощью операций сложения, умножения и суперпозиции из непрерывных функций одного переменного, что, собственно, и позволяют делать нейронные сети с каскадным соединением их элементов.

Класс функций, вычислимый с помощью нейронных сетей, замкнут относительно линейных операций. Действительно, пусть есть нейронные сети $S_1$, $S_2$, ..., $S_k$, которые вычисляют функции $F_1$, $F_2$, ..., $F_k$ от вектора входных сигналов x. Линейная комбинация $\alpha_0 + \alpha_1F_1 + \alpha_2F_2 +...+ \alpha_kF_k$ вычисляется сумматором с весами $\alpha_0$, $\alpha_1$,

$\alpha_2$, ..., $\alpha_k$, на вход которого подаются выходные сигналы сетей $S_1$, $S_2$, ..., $S_k$. Разница в числе тактов функционирования этих сетей до получения ответа легко компенсируется «линиями задержки», составленными из синапсов с единичным весом.

Кроме того, класс функций, вычислимый с помощью нейронных сетей, замкнут относительно унарной операции, осуществляемой нелинейным преобразователем сигнала, входящим в состав нейрона: если сеть S вычисляет функцию F, то, подавая выход этой сети на вход нелинейного преобразователя, получим на его выходе функцию $\varphi(F)$. Тем самым, по теореме Колмогорова, множество функций, вычислимых нейронными сетями с заданной непрерывной нелинейной характеристической функцией, плотно в пространстве непрерывных функций от входных сигналов.

Покажем, что если нет ограничений на способы соединения элементов нейронных сетей, то с их помощью можно сколь угодно точно аппроксимировать работу любого непрерывного автомата. Каждый автомат имеет несколько входов (n), несколько выходов (p) и конечный набор (s) параметров состояния. Он вычисляет (s + p) функций от (n + s) переменных. Аргументы этих функций – входные сигналы и текущие параметры состояния. Значения функций – выходные сигналы и параметры состояния на следующем шаге. Каждый такой автомат можно представить как систему из (s + p) более простых автоматов (рис. 4.6), которые вычисляют по одной функции от (n + s) переменных. Смена состояний достигается за счет того, что часть значений этих функций на следующем шаге становится аргументами – так соединены автоматы.

Таким образом, без потери общности можно рассматривать сеть автоматов как набор устройств, каждое из которых вычисляет функцию нескольких переменных $f(x_1,... ,x_n)$. Этот простой, но фундаментальный факт позволяет утверждать, что, поскольку нейронные сети позволяют с любой точностью вычислять произвольную непрерывную функцию $f(x_1,... ,x_n)$, то, следовательно, с

их помощью можно сколь угодно точно аппроксимировать функционирование любого непрерывного автомата.

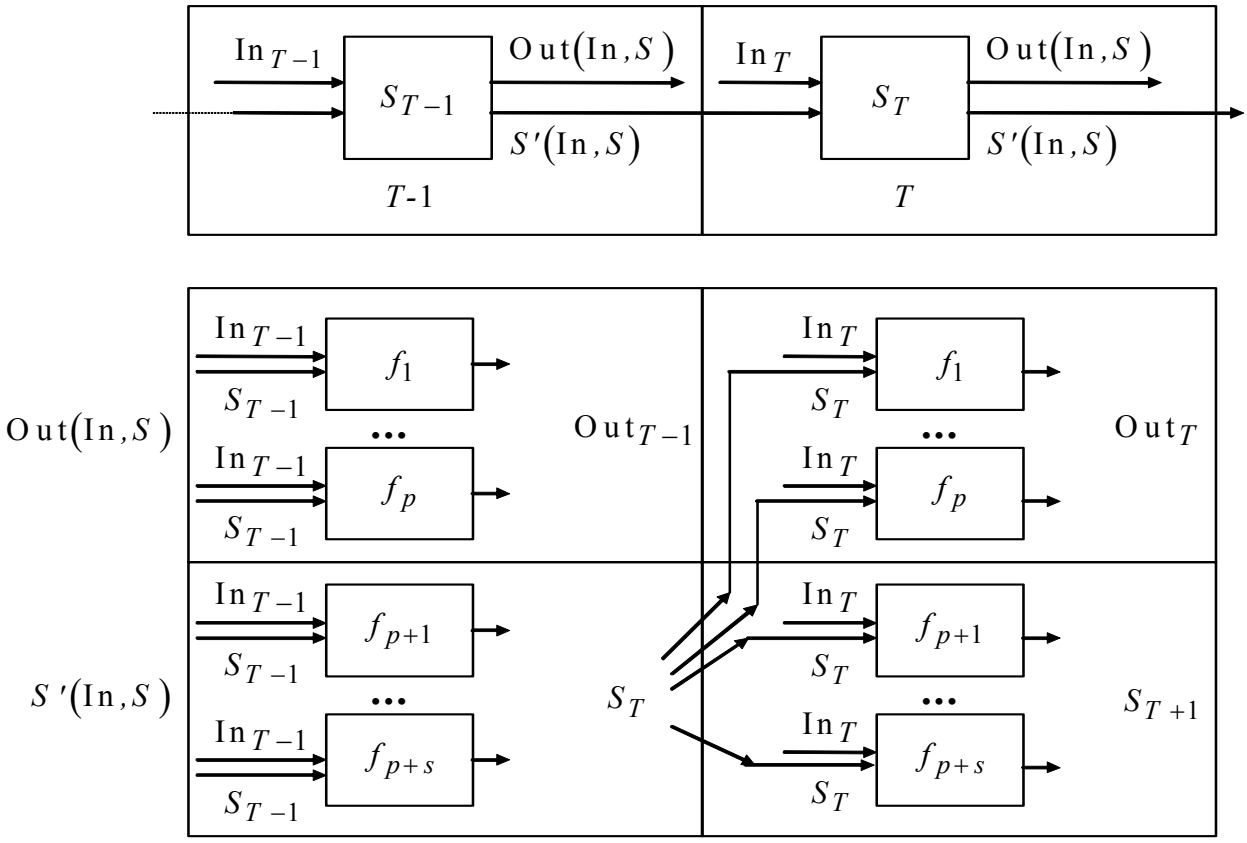


Рис. 4.6. *Представление автомата с помощью модулей, вычисляющих функции многих переменных от входных сигналов: на верхней схеме представлено функционирование автомата, на нижней он разложен на отдельные модули. Обозначения:*

*In – входные сигналы, Out – выходные сигналы, S – параметры состояния, T – дискретное время, Out(In,S) –- зависимость выходных сигналов от значений входных и параметров состояния, S′(In,S) – зависимость состояния в следующий момент дискретного времени от входных сигналов и текущего состояния, $f_1$-$f_p$ – функции переменных (In,S) – компоненты вектора Out(In,S), $f_{p+1}$-$f_{p+s}$ – функции переменных (In,S) – компоненты вектора S′(In,S), индексами T, T±1 обозначены соответствующие моменты времени.*

### 4.3.4. Поиск минимума квадратичного многочлена

Пусть $\alpha_{ij}$ – вес связи, ведущей от j-го нейрона к i-му. Для полносвязных сетей определены значения $\alpha_{ij}$ при всех i, j, для других архитектур связи, ведущие от j-го нейрона к i-му, для некоторых i, j не определены. В этом случае положим $\alpha_{ij}=0$. Далее речь пойдет в основном о полносвязных сетях.

Пусть на выходах всех нейронов получены сигналы $x_j$ (j – номер нейрона). Прохождение вектора сигналов x через сеть связей

сводится к умножению матрицы ($\alpha_{ij}$) на вектор сигналов x. В результате получаем вектор входных сигналов нелинейных элементов нейронов: $y_i = \sum_j \alpha_{ij} x_j$. Это соответствие (прохождение сети ≡ умножение матрицы связей на вектор сигналов) является основой для перевода обычных численных методов на нейросетевой язык и обратно. Практически всюду, где основной операцией является умножение матрицы на вектор, применимы нейронные сети. С другой стороны, любое устройство, позволяющее быстро осуществлять такое умножение, может использоваться для реализации нейронных сетей.

В частности, вычисление градиента квадратичной формы $H = \frac{1}{2}(x, Qx)$ может осуществляться полносвязной сетью с симметричной матрицей связей: $\mathrm{grad}H = Qx$ ($\alpha_{ij} = q_{ij} = q_{ji}$). Рассмотрим, как, используя нейросетевой подход, можно, например, методом наискорейшего спуска искать точку минимума многочлена второго порядка.

Пусть задан многочлен: $P(x) = \frac{1}{2}(x, Qx) + (b, x)$. Его градиент равен $\mathrm{grad}P = Qx + b$. Этот вектор может быть получен при прохождении вектора x через сеть с весами связей $\alpha_{ij} = q_{ij} = q_{ji}$ при условии, что на входной сумматора каждого нейрона по дополнительной связи веса b подается стандартный единичный сигнал.

Зададим теперь функционирование сети формулой

$$x' = x - h\,\mathrm{grad}P = x - h(Qx + b). \qquad (4.24)$$

Каждый j-й нейрон имеет входные веса $\alpha_{ij} = -hq_{ij}$ для связей с другими нейронами ($i \neq j$), вес $-b_j$ для постоянного единичного входного сигнала и вес $\alpha_{jj} = 1 - hq_{jj}$ для связи нейрона с самим собой (передачи на него его же сигнала с предыдущего шага). Выбор шага $h > 0$ может вызвать затруднение, он зависит от коэффи-

циентов минимизируемого многочлена. Есть, однако, простое решение: в каждый момент дискретного времени T выбирается свое значение $h_T$. Достаточно, чтобы шаг стремился со временем к нулю, а сумма шагов - к бесконечности (например $h_T = 1/T$, или $h_T = 1/\sqrt{T}$).

Итак, простая симметричная полносвязная сеть без нелинейных элементов может методом наискорейшего спуска искать точку минимума квадратичного многочлена.

### 4.3.5. Решение системы линейных уравнений

Как известно, решение уравнения $Ax = b$ сводится к минимизации многочлена

$$P = \frac{1}{2}((Ax - b),(Ax - b)) = \frac{1}{2}(x, A^T Ax) - (A^T b, x) - \frac{1}{2}(b, b). \quad (4.25)$$

Простейшая сеть, вычисляющая градиент этого многочлена, не полносвязна, а состоит из двух слоев: первый с матрицей связей $A$, второй – с транспонированной матрицей $A^T$. Постоянный единичный сигнал подается на связи с весами $b_j$ на первом слое. Минимизация этого многочлена, а значит и решение системы линейных уравнений, может проводиться так же, как и в общем случае, пошагово в соответствии с формулой $x' = x - h\,\mathrm{grad}P$.

Небольшая модификация позволяет вместо безусловного минимума многочлена второго порядка P искать точку условного минимума с условиями $x_i = c_i$ для $i = i_1,...,i_k$, то есть точку минимума P при ограничениях на аффинное многообразие, параллельное некоторым координатным плоскостям.

Для этого вместо формулы $x' = x - h\,\mathrm{grad}P = x - h(Qx + b)$ следует использовать:

$$x'_i = c_i \bullet p \text{ при } i = i_1,...,i_k;$$

$$x'_i = x_i - h\frac{\partial P}{\partial x_i} = x_i - h\left(\sum_j q_{ij}x_j + b_i\right) \bullet p \text{ при } i \neq i_1,...,i_k. \quad (4.26)$$

### 4.3.6. Восполнение данных

Предположим, что получаемый в ходе некоторого испытания вектор данных x подчиняется многомерному нормальному распределению:

$$\rho(x) = Ce^{-\frac{1}{2}((x-Mx),Q(x-Mx))}, \qquad (4.27)$$

где Mx – вектор математических ожиданий координат, $Q = \Sigma^{-1}$, $\Sigma$ – ковариационная матрица (ее оценка по выборке), n – размерность пространства данных, $C = \frac{1}{(2\pi)^{n/2}\sqrt{\det\Sigma}}$.

Напомним определение матрицы $\Sigma$:

$$(\Sigma)_{ij} = M((x_i - Mx_i)(x_j - Mx_j)), \qquad (4.28)$$

где M - символ математического ожидания.

В частности, простейшая оценка ковариационной матрицы по выборке дает:

$$S = \frac{1}{m}\sum_j (x^j - Mx) \otimes (Mx^j - Mx)^T, \qquad (4.29)$$

где m - число элементов в выборке, верхний индекс j - номер вектора данных в выборке, верхний индекс T означает транспонирование.

Пусть для вектора данных x известно несколько координат: $x_i = c_i$ для $i = i_1,...,i_k$. Наиболее вероятные значения неизвестных координат должны доставлять условный максимум показателю нормального распределения – многочлену второго порядка $((x-Mx),Q(x-Mx))$ (при условии $x_i = c_i$ для $i = i_1,...,i_k$). Эти же значения будут условными математическими ожиданиями неизвестных координат при заданных условиях.

Таким образом, чтобы построить сеть, заполняющую пробелы в данных, достаточно сконструировать сеть для поиска точек условного минимума многочлена $((x-Mx),Q(x-Mx))$ при услови-

ях следующего вида: $x_i = c_i$ для $i = i_1,...,i_k$. Матрица связей Q выбирается из условия $Q = \Sigma^{-1}$.

Пошаговое накопление $Q = \Sigma^{-1}$ по мере поступления данных требует слишком много операций - получив новый вектор данных, требуется пересчитать оценку $\Sigma$, а потом вычислить $Q = \Sigma^{-1}$. Можно поступать по-другому, воспользовавшись формулой приближенного обращения матриц первого порядка точности:

$$(\Sigma + \varepsilon\Delta)^{-1} = \Sigma^{-1} - \varepsilon\Sigma^{-1}\Delta\Sigma^{-1} + o(\varepsilon). \qquad (4.30)$$

Если же добавка $\Delta$ имеет вид $\Delta = y \otimes y^T$, то

$$(\Sigma + \varepsilon\Delta)^{-1} = \Sigma^{-1} - \varepsilon(\Sigma^{-1}y) \otimes (\Sigma^{-1}y)^T + o(\varepsilon). \qquad (4.31)$$

Заметим, что решение задачи не меняется при умножении Q на число. Поэтому полагаем:

$$Q_0 = 1, Q_{k+1} = Q_k + \varepsilon[Q_k(x^{k+1} - (Mx)^{k+1})] \otimes [Q_k(x^{k+1} - (Mx)^{k+1})]^T, \qquad (4.32)$$

где 1 – единичная матрица, $\varepsilon > 0$ – достаточно малое число, $x^{k+1}$ – k + 1-й вектор данных, $(Mx)^{k+1}$ – среднее значение вектора данных, уточненное с учетом $x^{k+1}$:

$$(Mx)^{k+1} = \frac{1}{k+1}(k(Mx)^k + x^{k+1}).$$

В формуле для пошагового накопления матрицы Q ее изменение $\Delta Q$ при появлении новых данных получается с помощью вектора $y = x^{k+1} - (Mx)^{k+1}$, пропущенного через сеть: $(\Delta Q)_{ij} = \varepsilon z_i z_j$, где z = Qy. Параметр $\varepsilon$ выбирается достаточно малым для того, чтобы обеспечить положительную определенность получаемых матриц, и, по возможности, их близость к истинным значениям Q.

Описанный процесс формирования сети можно назвать ее обучением. Если при обучении сети поступают некомплектные данные $x^{k+1}$ с отсутствием значений некоторых координат, то сначала эти значения восстанавливаются с помощью имеющейся се-

ти, а потом используются в ее дальнейшем обучении. Циклическое функционирование сети следует прекратить, то есть, остановиться и считать полученный результат ответом, если изменения выходных сигналов сети за цикл, меньше некоторого фиксированного малого $\delta$ (при использовании переменного шага $\delta$ может быть его функцией).

### 4.3.7. Ассоциативная память

До сих пор речь шла о минимизации положительно определенных квадратичных форм и многочленов второго порядка. Однако самое интересное приложение полносвязных нейронных сетей связано с увеличением значений положительно определенных квадратичных форм. Речь идет о системах ассоциативной памяти [Кохонен, 1980, 1982]. Предположим, что задано несколько эталонных векторов данных $x^1,...,x^m$ и при обработке поступившего на вход системы вектора x требуется получить на выходе ближайший к нему эталонный вектор. Мерой сходства в простейшем случае будем считать косинус угла между векторами – для векторов фиксированной длины это просто скалярное произведение. Можно ожидать, что изменение вектора x происходит по закону

$$x' = x + h\sum_k x^k (x^k, x), \qquad (4.33)$$

где h – малый шаг, приведет к увеличению проекции x на те эталоны, скалярное произведение на которые $(x^k, x)$ больше.

Ограничимся рассмотрением эталонов, и ожидаемых результатов обработки с координатами $\pm 1$. Развивая изложенную идею, приходим к дифференциальному уравнению

$$\frac{dx}{dt} = -\operatorname{grad} H,$$

$$H = H_0 + \theta H_1,\ H_0(x) = -\frac{1}{2}\sum_k (x^k, x)^2,\ H_1(x) = \frac{1}{2}\sum_i (x_i^2 - 1)^2, \qquad (4.34)$$

$$\operatorname{grad} H_0 = -\sum_k x^k (x^k, x),\ (\operatorname{grad} H_1)_j = (x_j^2 - 1)x_j,$$

где верхними индексами обозначаются номера векторов-эталонов, нижними - координаты векторов.

Функция H называется «энергией» сети, она минимизируется в ходе функционирования. Слагаемое $H_0$ вводится для того, чтобы со временем возрастала проекция вектора x на те эталоны, которые к нему ближе, слагаемое $H_1$ обеспечивает стремление координат вектора x к $\pm 1$. Параметр $\theta$ определяет соотношение между интенсивностями этих двух процессов. Целесообразно постепенно менять $\theta$ со временем, начиная с $\theta < 1$, и приходя в конце концов к $\theta > 1$.

Матрица связей построенной сети определяется функцией $H_0$, так как $(\mathrm{grad}H_1)_j = (x_j^2 - 1)x_j$ вычисляется непосредственно при j-м нейроне без участия сети. Вес связи между i-м и j-м нейронами не зависит от направления связи и равен

$$\alpha_{ij} = \sum_k x_i^k x_j^k . \qquad (4.35)$$

Эта простая формула имеет чрезвычайно важное значение для развития теории нейронных сетей. Вклад k-го эталона в связь между i-м и j-м нейронами ($x_i^k x_j^k$) равен +1, если i-я и j-я координаты этого эталона имеют одинаковый знак, и равен -1, если они имеют разный знак.

В результате возбуждение i-го нейрона передается j-му (и симметрично, от j-го к i-му), если у большинства эталонов знак i-й и j-й координат совпадают. В противном случае эти нейроны тормозят друг друга: возбуждение i-го ведет к торможению j-го, торможение i-го - к возбуждению j-го (воздействие j-го на i-й симметрично). Это правило образования ассоциативных связей (правило Хебба) сыграло важную роль в теории нейронных сетей.

### 4.3.8. Кластер-анализ и классификации без учителя

Построение отношений на множестве объектов – одна из самых открытых для творчества областей применения нейросетевых моделей. Первым и наиболее распространенным примером этой задачи является классификация без учителя. Задан набор объектов, каждому из которых сопоставлен вектор значений признаков (строка таблицы). Требуется разбить эти объекты на классы эквивалентности. Отнесение объекта к классу проводится путем его сравнения с типичными элементами разных классов и выбора ближайшего.

Простейшая мера близости объектов – квадрат евклидового расстояния между векторами значений их признаков (чем меньше расстояние – тем ближе объекты). Соответствующее определение признаков типичного объекта – среднее арифметическое значение признаков по выборке, представляющей класс. Другая мера близости, естественно возникающая при обработке сигналов, изображений и т. п. – квадрат коэффициента корреляции (чем он больше, тем ближе объекты). Если число классов m заранее определено, то формально задачу классификации без учителя можно выразить следующим образом.

Пусть $\{x^p\}$ – векторы значений признаков для рассматриваемых объектов, в пространстве которых определена мера их близости $\rho\{x, y\}$. Для определенности примем, что чем ближе объекты, тем меньше $\rho$. С каждым классом будем связывать его типичный объект – ядро класса. Требуется определить набор из m ядер $y^1$, $y^2$, ... $y^m$ и разбиение $\{x^p\}$ на классы: $\{x^p\} = Y_1 \cup Y_2 \cup ... \cup Y_m$, минимизирующее следующий критерий

$$Q = \sum_{i=1}^{m} D_i \to \min, \qquad (4.36)$$

где для каждого (i-го) класса $D_i$ – сумма расстояний от принадлежащих ему точек выборки до ядра класса:

$$D_i = \sum_{x^p \in Y_i} \rho(x^p, y^i). \qquad (4.37)$$

Минимум Q берется по всем возможным положениям ядер $y^i$ и всем разбиениям $\{x^p\}$ на m классов $Y_i$.

Если число классов заранее не определено, то полезен критерий слияния классов: классы $Y_i$ и $Y_j$ сливаются, если их ядра ближе, чем среднее расстояние от элемента класса до ядра в одном из них. Возможны варианты использования среднего расстояния по обоим классам, или порогового коэффициента, показывающего, во сколько раз расстояние между ядрами должно превосходить среднее расстояние от элемента до ядра.

Использовать критерий слияния классов можно так: сначала принимаем гипотезу о достаточном числе классов; строим их, минимизируя Q; затем некоторые $Y_i$ объединяем; повторяем минимизацию Q с новым числом классов и т.д.

Сетевые алгоритмы классификации без учителя строятся на основе итерационного метода динамических ядер. Опишем его сначала в наиболее общей абстрактной форме. Пусть задана выборка предобработанных векторов данных $\{x^p\}$. Пространство векторов данных обозначим E. Каждому классу будет соответствовать некоторое ядро a. Пространство ядер будем обозначать A. Для каждых $x \in E$ и $a \in A$ определяется мера близости d(x, a). Для каждого набора из k ядер $a_1,..., a_k$ и любого разбиения $\{x^p\}$ на k классов $\{x^p\} = P_1 \cup P_2 \cup ... \cup P_k$ зададим критерий качества:

$$D = D(a_1, a_2, ..., a_k, P_1, P_2, ...P_k) = \sum_{i=1}^{k} \sum_{x \in P_i} d(x, a_i). \qquad (4.38)$$

Требуется найти набор $a_1,..., a_k$ и разбиение $\{x^p\} = P_1 \cup P_2 \cup ... \cup P_k$, минимизирующие D.

Шаг алгоритма разбивается на два этапа:

1-й этап – для фиксированного набора ядер $a_1,..., a_k$ ищем разбиение $\{x^p\} = P_1 \cup P_2 \cup ... \cup P_k$, минимизирующее критерий каче-

ства D; оно дается решающим правилом: $x \in P_i$, если $d(x, a_i) < d(x, a_j)$ при $i \neq j$, в том случае, когда для x минимум d(x, a) достигается при нескольких значениях i, выбор между ними может быть сделан произвольно;

2-й этап – для каждого $P_i$ (i = 1,..., k), полученного на первом этапе, ищется $a_i \in A$, минимизирующее критерий качества (то есть слагаемое в D для данного i - $D_i = \sum_{x \in P_i} d(x, a_i)$.

Начальные значения $a_1,..., a_k$, $\{x^p\} = P_1 \cup P_2 \cup ... \cup P_k$ выбираются произвольно, либо исходя из сущности решаемой задачи.

На каждом шаге и этапе алгоритма уменьшается критерий качества D, отсюда следует сходимость алгоритма - после конечного числа шагов разбиение $\{x^p\} = P_1 \cup P_2 \cup ... \cup P_k$ уже не меняется. Если ядру $a_i$ сопоставляется элемент сети, вычисляющий по входному сигналу x функцию $d(x, a_i)$, то решающее правило для классификации выглядит следующим образом: элемент x принадлежит классу $P_i$, если выходной сигнал i-го элемента $d(x, a_i)$ меньше всех остальных.

Единственная вычислительная сложность описанного алгоритма состоит в поиске ядра по классу на втором этапе алгоритма, то есть в поиске $a \in A$, минимизирующего $D_i = \sum_{x \in P_i} d(x, a)$. В связи с этим, в большинстве конкретных реализаций данного алгоритма мера близости d выбирается такой, чтобы можно было легко найти a, минимизирующее D для данного P .

В простейшем случае пространство ядер A совпадает с пространством векторов x, а мера близости d(x, a) – положительно определенная квадратичная форма от x – a может быть представлена, например, как квадрат евклидового расстояния. Тогда ядро $a_i$, минимизирующее $D_i$, есть центр тяжести класса $P_i$:

$$a_i = \frac{1}{|P_i|} \sum_{x \in P_i} x, \qquad (4.39)$$

где $|P_i|$ – число элементов в $P_i$.

В этом случае упрощается и решающее правило, разделяющее классы. Обозначим d(x, a) = (x – a, x – a), где (.,.) – билинейная форма (если d – квадрат евклидового расстояния между x и a, то (.,.) – обычное скалярное произведение). В силу билинейности

$$d(x, a) = (x - a, x - a) = (x, x) - 2(x, a) + (a, a). \qquad (4.40)$$

Чтобы сравнить $d(x, a_i)$ для разных i и найти среди них минимальное, достаточно вычислить линейную неоднородную функцию от x: $d_1(x, a_i) = (a_i, a_i) - 2(x, a_i)$. Минимальное значение $d(x, a_i)$ достигается при том же i, что и минимум $d_1(x, a_i)$, поэтому решающее правило реализуется с помощью k сумматоров, вычисляющих d(x, a) и интерпретатора, выбирающего сумматор с минимальным выходным сигналом. Номер этого сумматора и есть номер класса, к которому относится x.

Пусть теперь мера близости – коэффициент корреляции между вектором данных и ядром класса:

$$d(x, a) = r(x, a) = \sum_j \frac{(x_j - M_x)(a_j - M_a)}{\sigma_x \sigma_a}, \qquad (4.41)$$

где $x_j, a_j$ – координаты векторов, $M_x = \frac{1}{n}\sum_j x_j$ (аналогично для $M_a$), n – размерность пространства данных,

$\sigma_x = \sqrt{\frac{1}{n}\sum_j (x_j - M_x)^2}$ (аналогично для $\sigma_a$).

Предполагается, что данные предварительно обрабатываются (нормируются и центрируются) по правилу:

$$x \to \frac{x_j - M_x}{\sigma_x}. \qquad (4.42)$$

Точно также нормированы и центрированы векторы ядер a. Поэтому все обрабатываемые векторы и ядра принадлежат сечению единичной евклидовой сферы (||x|| = 1) гиперплоскостью

($\sum_i x_i = 0$). В таком случае $d(x, a) = (x, a)$ и задача поиска ядра для данного класса P имеет своим решением

$$a_P = \frac{\sum_{x \in P} x}{\left\| \sum_{x \in P} x \right\|}. \qquad (4.43)$$

В описанных простейших случаях, когда ядро класса точно определяется как среднее арифметическое (или нормированное среднее арифметическое) элементов класса, а решающее правило основано на сравнении выходных сигналов линейных адаптивных сумматоров, нейронную сеть, реализующую метод динамических ядер, называют сетью Кохонена. В определении ядер a для сетей Кохонена входят суммы $\sum_{x \in P} x$. Это позволяет накапливать новые динамические ядра, обрабатывая по одному примеру и пересчитывая $a_i$ после появления в $P_i$ нового примера. Сходимость при такой модификации, однако, ухудшается.

### 4.3.9. Нечеткая классификация

Пусть вектор данных x обработан слоем элементов нейросети, вычисляющих $y_i = d(x, a_i)$. Идея дальнейшей обработки состоит в том, чтобы выбрать из этого набора $\{y_i\}$ несколько самых больших чисел и после нормировки объявить их значениями функций принадлежности к соответствующим классам. Предполагается, что к остальным классам объект наверняка не принадлежит. Для выбора семейства G наибольших $y_i$ определим следующие числа:

$$y_{max} = \max\{y_i\}, M_y = \frac{1}{k}\sum_i y_i, s = (1-\alpha)M_y + \alpha y_{max} \qquad (4.44)$$

где число $\alpha$ характеризует отклонение s от своего среднего значения $M_y$, $\alpha \in [-1, 1]$.

Множество $J = \{i|y_i \in G\}$ трактуется как совокупность номеров тех классов, к которым может принадлежать объект, а нормированные на единичную сумму неотрицательные величины

$$f_i = \frac{y_i - s}{\sum_{j \in J}(y_i - s)} \text{ (при } i \in J) \text{ и } f = 0 \text{ (в противном случае),}$$

интерпретируются как значения функций принадлежности этим классам.

С каждым классом связывается q-мерный выходной вектор $z^i$. Строится слой из q выходных сумматоров, каждый из которых должен выдавать свою компоненту выходного вектора. Весовые коэффициенты связей, ведущих от того элемента нечеткого классификатора, который вычисляет $f_i$, к j-му выходному сумматору определяются как $z^i_j$. В итоге вектор выходных сигналов сети есть $z = \sum_i f_i z^i$ .

* * *

Мы рассмотрели основные принципы построения нейросетевых моделей и лишь частично затронули некоторые области их практического использования. Полагаем, что изложенного вполне достаточно для того, чтобы системный аналитик (не специалист в области нейроинформатики) получил представление об этом сравнительно новом, но бурно развивающимся подходе к моделированию сложных систем

С точки зрения моделирования систем можно выделить три неоспоримых преимущества нейросетевого подхода. Во-первых, нейронные сети обладают способностью быстрой обработки информации, что дает возможность расширить набор аналитических операций, выполняемых в реальном режиме времени. Во-вторых, нейронные сети способны к обучению, а уже обученные они могут эффективно обобщать полученную информацию и демонстрировать хорошие результаты на данных, не использовавшихся в

процессе обучения. Это качество позволяет использовать нейросетевые модели для имитации функционирования систем, для которых не удается установить их состав, структуру и принципы работы, но частично известны реакции «вход-выход». В-третьих, однородность, проявляющаяся в составе и конструкции элементов сети, обеспечивает возможность построения нейросетевых моделей на базе однотипных аппаратных или программных средств, что существенно ускоряет процесс их проектирования и реализации.

С точки зрения практического приложения нейросетевых моделей наиболее важным является вопрос о том, способны ли они расширить диапазон прогноза изучаемых явлений и, прежде всего, конфликтных. Речь идет не о демонстрационных примерах (типа предсказания результатов выборов президента), а о фундаментальных механизмах, позволяющих с помощью этого класса моделей более или менее достоверно прогнозировать конфликтное будущее, базируясь на материале прошлого и настоящего. Убедительного ответа на этот вопрос пока (пока ли?) не получено.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

*Аверкин А.Н., Батыршин И.З. и др.* Нечеткие множества в моделях управления и искусственного интеллекта. / Под ред. *Поспелова Д.А.* – М., 1986.

*Акоф Ф.Р.* Искусство решения проблем. – М.: Сов. радио, 1982.

*Акоф Ф.Р., Эмери Ф.О.* О целеустремленных системах. – М.: Сов. радио, 1974.

*Альянах И.Н.* Моделирование вычислительных систем. – Л.: Машиностроение. Ленинград, отд-ние, 1988.

*Анализ сложных систем* / Под ред. *Э. Квейда*. – М.: Наука, 1969.

*Аржакова Н.В., Новосельцев В.И., Редкозубов С.А.* Управление динамикой рынка: системный подход. – Воронеж: ВГУ, 2004.

*Баркалов С.А., Бурков В.Н., Гилязов Н.М., Семенов П.И.* Минимизация упущенной выгоды в задачах управления проектами. – М.: ИПУ РАН, 2001.

*Беллман Р*. Динамическое программирование. – М.: Наука, 1960.

*Брябрин В.М. и др.* Диалоговые системы в АСУ. – М.: Машиностроение, 1983.

*Бурков В.Н.* Основы математической теории активных систем. – М.: Наука, 1977, – 225с.

*Гайкович Ю.А., Тарасов Б.В.* Интеллектуальные средства ввода, вывода и обработки речевой и графической информации в АСУ. М.: Воениздат, 1989.

*Гермейер Ю.В*. Игры с непротиворечивыми интересами. – М., 1976.

*Голиков В.К., Матусов К.Н., Сысоев В.В.* Сети Петри в ситуационном управлении и имитационном моделировании дискретных технологических систем / Под общ. ред. *В.В. Сысоева.* – М.: ИПРЖР, 2002.

*Goldberg D.E., Korb B., Deb K.* Messy genetic algorithms: Motivation, analysis, and first results. Complex Systems, 3, pp. 493-530, 1989.

*Горбань А.Н.* Обучение нейронных сетей. М.: изд. СССР–США СПб "ПараГраф", 1990.

*Горбань А.Н., Россиев Д.А.* Нейронные сети на персональном компьютере. Новосибирск: Наука (Сиб. отделение), 1996.

*Гладун В.П.* Эвристический поиск в сложных средах. – Киев: Наукова думка, 1977.

*Данциг Д.* Линейное программирование. – М.: Прогресс, 1960.

*Дёмин Б.Е. и др.* Пояснительная записка к техническому проекту создания модернизированного варианта Государственной автоматизированной системы Российской Федерации «Выборы» (ГАС «Выборы»). – М.: НИИ «Восход», 2002.

*Дёмин Б.Е., Клочков В.В.* Оценка эффективности общесистемных решений усовершенствованной ГАС «Выборы» и использование научно-технического потенциала для дальнейшего развития системы в 2005-2008 годы. «Информатизация и связь», № 4, 2004.

*De Jong.* Analysis of the.behaviour of a class of genetic adaptive systems, PhD thesis, Untv. of Michigan, 1975.

*Дорожкин В.Р., Гасилов В.В., Баркалов С.А.* Подрядные торги в строительстве: Уч. пособие / под ред. В.Р. Дорожкина. – Воронеж: ВГАСУ, 2003.

*Дынкин Е.Б., Юшкевич А.А.* Теоремы и задачи о процессах Маркова. – М.: Наука, 1967.

*Егоров В.А., Лузанов В.Д., Щербаков С.М.* Транспортно–накопительные системы для ГПС – Л.: Машиностроение. Ленингр. отд-ние, 1989.

*Заде Л.* Основы нового подхода к анализу сложных систем и процессов принятия решений (Новое в жизни, науке, технике. Серия «Математика и кибернетика», №.7). – М., 1974.

*Заде Л.* Понятие лингвистической переменной и его применение к принятию приближенных значений. – М., 1976.

*Квейд Э.* Методы системного анализа. – В кн.: Новое в теории и практике управления производством США. – М., 1971.

*Колмогоров А.Н.* О представлении непрерывных функций нескольких переменных в виде суперпозиции непрерывных функций одного переменного. Докл. АН СССР, 1957. Т. 114, No. 5. с. 953-956.

*Кохонен Т.* Ассоциативная память. – М.: Мир, 1980.

*Кохонен Т.* Ассоциативные запоминающие устройства. – М.: Мир, 1982.

*Клыков Ю.И.* Ситуационное управление большими системами. – М.: Энергия, 1974.

*Крон Г.* Исследование сложных систем по частям – диакоптика. – М.: Наука, 1972.

*Крушанов А.А.* К вопросу о природе управления. В кн.: Информация и управление. Философско-методологические проблемы. – М.: Наука, 1985.

*Кузнецов В.И.* Системное проектирование радиосвязи. Часть 1 (Системотехника). – Воронеж: ВНИИС, 1994.

*Кузнецов В.И.* Системное проектирование радиосвязи. Часть 2 (Обеспечение) и часть 3 (Планирование и управление). – Воронеж: ВНИИС, 2000.

*Левиатов А.Ю., Захаров В.Н.* Непрямые методы диагностики // Международный симпозиум по искусственному интеллекту. – Л.: ISAI, 1983. – С.67-72.

*Lorenz E.N.* Deterministic nonperiodic flow // Journ. of the Atmospheric Science. 1963. V. 20. P. 130-141.

*Липаев В.В.* Обеспечение качества программных средств. – М.: СИНТЕГ, 2001.

*Липаев В.В.* Технико-экономическое обоснование проектов сложных программных средств. М.: СИНТЕГ, 2004.

*Макаров С.О.* Рассуждения по вопросам морской тактики. – М.: Военмориздат, 1942.

*Моисеев Н.Н.* Элементы теории оптимальных систем. – М., 1975.

*Минский М., Пайперт С.* Персептроны. – М.: Мир, 1971.

*Minsky M.* A. Framework for Representing Knowledge, in The Psychology of Computer Vision, P.H. Winston (ed.), McGraw-Hill, 1975.

*Мухачева Э.А., Рубинштейн Г.Ш.* Математическое программирование. – Новосибирск: Наука, 1977.

*Нейман Дж. фон, Моргенштерн О.* Теория игр и экономическое поведение. – М., 1970.

Нечеткие множества и теория возможностей. Последние достижения / Перевод с англ. под ред. *Р.Р. Ягера.* – М., 1986.

*Новосельцев В.И.* Системный анализ: современные концепции / изд. 2-е испр. и дополн. – Воронеж: Кварта, 2003.

*Новосельцев В.И.* Системная конфликтология. – Воронеж: Кварта, 2001.

*Новосельцев В.И., Тарасов Б.В. и др.* Логико-лингвистические модели в военных системных исследованиях. – М.: Воениздат, 1988.

*Оптнер С.Л.* Системный анализ для решения деловых и промышленных проблем. – М.: Мир, 1969.

*Питерсон Дж.* Теория сетей Петри и моделирование систем / пер. с англ. – М.: Мир, 1984.

*Плаус С.* Психология оценки и принятия решений. – М., 1998, с. 289.

*Подиновский В.В., Ногин В.Д.* Парето-оптимальные решения многокритериальных задач. – М.: Наука, 1982.

*Понтрягин Л.С. и др.* Математическая теория оптимальных процессов. – М.: Наука, 1976.

*Поспелов Д.А.* Большие системы: Ситуационное управление. – М., 1975.

*Поспелов Д.А.* Логико-лингвистические модели в системах управления. – М., 1981.

*Райфа Г.* Анализ решений. – М.: Наука, 1977.

*Рейуорд-Смит В. Дж.* Теория формальных языков. Вводный курс. – М., 1988.

*Розенблатт Ф.* Принципы нейродинамики. Перцептрон и теория механизмов мозга. М.: Мир, 1965

*Рындин А.А., Хаустович А.В., Долгих Д.В., Мугалев А.И., Сапегин С.В.* Проектирование корпоративных информационных систем. – Воронеж: Кварта, 2003.

*Самарский А.А., Михайлов А.П.* Математическое моделирование. Идеи. Методы. Примеры. – М., Наука, 1997.

*Сысоев В.В.* Автоматизированное проектирование линий и комплексов оборудования полупроводникового и микроэлектронного производства. – М.: Радио и связь, 1982.

*Сысоев В.В.* Конфликт. Сотрудничество. Независимость. Системное взаимодействие в структурно-параметрическом представлении. – М., 1999.

Технология системного моделирования / Е.Ф. Аврамчук, А.А, Вавилов, С.В. Емельянов и др. – М.: Машиностроение; – Берлин: Техник, 1988

*Уемов А.И.* Системный подход и общая теория систем. – М., 1978.

*Уидроу Б., Стирнз С.* Адаптивная обработка сигналов. М.: Мир, 1989.

*Уинстон П.* Искусственный интеллект. – М., 1980.

*Уэно Х., Исидзука М.* Представление и использование знаний. – М., 1989.

*Хакен Г.* Синергетика: иерархия неустойчивостей в самоорганизующихся системах и устройствах. – М.: Мир, 1985.

*Хакен Г.* Информация и самоорганизация: Макроскопический подход к сложным системам. – М.: Мир, 1991.

*Хитч Ч.* Руководство обороной. – М., 1968.

*Xiaofeng Q, Palmieti F.* Theoretical analysis of evolutionary algorithms ijlth an infinite population size in continuous space. Parts 1, 11, IEEE Trans. on Neural Networks, Vol.5, No.1, 102-130, 1994.

*Harti R.E.* A global convergence proof for class of genetic algorithms. Technische Universitat Wien, 1990.

*Holland J.H.* Adaptive plans optimal for payoff-only environments, Proc. of the 2nd Hawaii Int. Conf. on System Sciences, pp. 917-920, 1969.

*Holland J.H.* Adaptation in Natural and Artificial Systems. Ann Arbor: Univ. of Michigan Press, 1975.

*Элти Дж., Кумбс М.* Экспертные системы: концепции и примеры. – М., 1987.

*Эсаулов А.Ф.* Психология решения задач. – М.: Высшая школа, 1972

*Эшби У.* Несколько замечаний. – Общая теория систем. – М.: Мир-, 1966.

*Яцук В.Я.* Использование λ-фреймов и метода продукций при построении интеллектуальных систем принятия решений // Международный симпозиум по искусственному интеллекту. – Л., 1982.

Научное издание

АРЖАКОВ Михаил Владимирович
АРЖАКОВА Наталия Владимировна
ГОЛИКОВ Виктор Константинович
ДЁМИН Борис Евгеньевич
НОВОСЕЛЬЦЕВ Виктор Иванович

# МОДЕЛИРОВАНИЕ СИСТЕМ

Ответственный редактор В.Ю. Колчев

Подписано в печать 20.08.05. Формат 60×84 1/16.
Бумага офсетная. Гарнитура Ньютон. Ризография.
Усл. печ. л. 10,2. Тираж 500 экз. Заказ 125.