

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ
КАФЕДРА КОМПЬЮТЕРНЫХ СИСТЕМ В УПРАВЛЕНИИ
И ПРОЕКТИРОВАНИИ

А.Г. Карпов

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ТЕОРИИ СИСТЕМ

Часть 1

Учебное пособие



Томск – 2007

УДК 681.513.2.01
К 265

Карпов А.Г. Математические основы теории систем. Часть 1:
К 265 Учебное пособие. – Томск: Изд-во НТЛ, 2007. – 184 с.

ISBN 978-5-89503-357-9

В учебном пособии даны общие определения, термины и понятия теории систем и системного анализа. В первой части приведено математическое описание самого простого класса систем – систем, имеющих конечное множество значений переменных, характеризующих поведение этих систем. Таким математическим описанием является теория конечных автоматов. Рассмотрены также вопросы анализа и синтеза автоматов, как на абстрактном, так и на структурном уровнях.

Учебное пособие предназначено для студентов любых форм обучения, в том числе и с применением дистанционных образовательных технологий

УДК 681.513.2.01

Рецензент: профессор Томск. гос. ун-та систем управления и радиоэлектроники **А.Г. Гарганеев**

ISBN 978-5-89503-357-9

© А.Г.Карпов, макет, 2007
© Том. гос. ун-т систем упр. и радиоэлектроники, 2007

СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ.....	6
1. ВВЕДЕНИЕ. ОСНОВНЫЕ ПОНЯТИЯ	7
1.1. Предварительные замечания.....	7
1.2. Системность человеческой практики	8
1.3. Системность познавательных процессов и природы	10
1.4. Общие свойства систем	11
2. МОДЕЛИ И МОДЕЛИРОВАНИЕ	14
2.1. Понятие модели и его развитие	14
2.2. Типы моделей	17
2.2.1. Модели прагматические и познавательные	17
2.2.2. Модели статические и динамические	18
2.2.3. Абстрактные модели.....	18
2.2.4. Материальные модели	20
2.3. Свойства моделей.....	22
2.3.1. Условия реализации моделей	22
2.3.2. Различия между моделью и оригиналом	22
2.3.3. Сходство между моделью и оригиналом	24
3. СИСТЕМЫ, ИХ ОБЩЕЕ ОПИСАНИЕ И КЛАССИФИКАЦИЯ	27
3.1. Первое определение системы. Модель "Чёрный ящик"	27
3.2. Модель состава системы	30
3.3. Модель структуры системы. Второе определение системы.....	31
3.4. Динамические модели системы	32
3.4.1. Типы динамических моделей	33
3.4.2. Общая математическая модель динамической системы	35
3.5. Классификация систем	41
3.5.1. Классификация по происхождению.....	42
3.5.2. Типы операторов систем	45

3.5.3. Типы способов управления.....	46
3.5.4. Ресурсное обеспечение.....	48
4. АВТОМАТНОЕ ОПИСАНИЕ СИСТЕМ. ТЕОРИЯ КОНЕЧНЫХ АВТОМАТОВ.....	50
4.1. Основные понятия. Способы задания автоматов	50
4.1.1. Определение абстрактного автомата	50
4.1.2. Задание автоматов.....	55
4.2. Виды автоматов и их свойства	58
4.2.1. Автономные автоматы.....	58
4.2.2. Автоматы синхронные и асинхронные.....	61
4.2.3. Автоматы Мили и автоматы Мура.....	63
4.2.4. Автоматы первого и второго рода	67
4.2.5. Гомоморфизм, изоморфизм и эквивалентность автоматов	71
4.2.6. Минимизация автоматов	73
4.2.7. Частичные автоматы и их свойства	77
4.3. Распознавание множеств автоматами	84
4.3.1. Понятие события и постановка задачи представления событий автоматами	84
4.3.2. Регулярные события и алгебра Клини.....	88
4.3.3. Синтез автоматов (абстрактный уровень).....	95
4.3.4. Анализ автоматов (абстрактный уровень)	100
4.4. Алгебра абстрактных автоматов	106
4.4.1. Теоретико-множественные операции	107
4.4.2. Алгебраические операции.....	110
4.4.3. Операции над вероятностными автоматами	132
4.5. Структурное исследование автоматов	141
4.5.1. Комбинационные логические автоматы.....	142
4.5.2. Постановка задач синтеза и анализа на структурном уровне	143
4.5.3. Элементный базис.....	144

4.5.4. Автоматные сети	147
4.5.5. Анализ комбинационных автоматов.....	155
4.5.6. Синтез комбинационных автоматов	157
4.5.7. Кодирование состояний	164
4.5.8. Программная реализация комбинационных автоматов.....	170
4.6. Общие методы синтеза автоматов.....	174
4.6.1. Декомпозиция абстрактных автоматов	175
4.6.2. Канонический метод синтеза.....	177
4.6.3. Декомпозиционный метод синтеза	179
СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ	181

ПРЕДИСЛОВИЕ

Курс "Математические основы теории систем" предваряет изучение таких дисциплин как "Теория управления", "Цифровые системы автоматического управления", "Технические средства автоматизации и управления", "Моделирование систем управления " и некоторые другие. В этом курсе продолжается углубленное изучение тех разделов математики, которые непосредственно связаны с описанием и исследованием динамических систем. Ввиду большого объема курс лекций разделен на две части.

В первой части даются общие определения, термины и понятия теории систем и системного анализа и приводится так называемое автоматное описание систем, которое пригодно для исследования любых цифровых измерительных, управляющих или вычислительных систем.

1. ВВЕДЕНИЕ. ОСНОВНЫЕ ПОНЯТИЯ

1.1. Предварительные замечания

О системах, "системности", "системном подходе", "системном анализе", "теории систем" и т.п. пишут и говорят часто. Много из того, что вчера называли единым, комплексным, целостным и т.п., сегодня называют модным теперь словечком "*системный*". Но за этой модой, и это подчеркивают не только ученые, но и инженеры, педагоги, организаторы производства, деятели культуры и другие, стоит широкое осознание системности как одной из важных характеристик окружающего нас мира и осмысления ее как особого измерения этого мира.

Широко распространилось понятие, что наши успехи и неудачи связаны с тем, насколько системно мы подходим к решению проблемы.

Неверно считать, что мышление стало системным в последнее время. Оно было таким всегда и другим быть не может. Системность – это не такое качество, которым можно обладать или нет. Она имеет разные уровни. Сигналом о недостаточной системности служит появление проблемной ситуации: разрешение возникшей проблемы осуществляется переходом на новый, более высокий уровень системности в нашей деятельности. Поэтому это не столько состояние, сколько процесс.

Если вам не совсем ясно, понятно, представляется расплывчатым, что означает само слово "*система*"; "*действовать системно*", почему не-системного знания не бывает – то это как раз пример возникновения проблемной ситуации. Есть проблема понимания сказанного, которую мы будем решать, постепенно повышая *уровень системности знаний*.

Осознание системности мира пришло не сразу и с трудом. Но оно не могло не прийти. Системные представления возникли по объективным причинам и развиваются под действием объективных факторов.

Далее мы попытаемся проследить за развитием системности в практической деятельности, в мышлении и, вообще, в окружающем нас мире.

1.2. Системность человеческой практики

Что является наиболее важным и самым интересным для человечества? Ну, например, можно на это ответить так: это вопросы возможности человека в его отношениях с природой и с самим собой, способы реализации этих возможностей, факторы, способствующие и препятствующие расширению этих возможностей. Здесь же философская проблема соотношения материи и сознания.

Возьмем практическую деятельность человека, то есть его активное и целенаправленное воздействие на окружающую среду. Оно системно. Позже рассмотрим все признаки системности, а сейчас пришло время отметить только самое необходимое и очевидное:

- структурированность системы,
- взаимосвязанность составляющих ее частей,
- подчиненность организации всей системы определенной цели.

По отношению к человеческой деятельности эти признаки очевидны:

- всякое осознанное действие преследует определенную цель;
- во всяком действии можно выделить составные части;
- эти составные части выполняются не произвольно, а в определенном порядке.

Это и есть определенная, подчиненная цели взаимосвязь составных частей.

Можно встретить и другое менее удачное название для такого построения деятельности: *алгоритмичность*. От алгоритма в математическом смысле здесь осталось только расчлененность на части этих шагов.

Следует учитывать, что роль системных представлений в практике постоянно растет.

В качестве примера можно привести повышение производительности труда. Низший уровень системности: *механизация*. Естественный предел механизации – сам человек, который управляет работой механизмов. Следующий уровень системности – *автоматизация*. В этом случае вообще исключаем человека из производственного процесса: на машины возлагаем не только выполнение непосредственно самой работы, но и выполнение операций по регулировке хода, течению процесса работы: автопилот, самонаведение ракеты, станок с числовым программным управлением, автоматические линии. Особое место здесь, конечно, играют ЭВМ. Однако полностью возлагать на машину можно только те работы, которые детально изучены, подробно и полно описаны, точно известно, что именно, в каком порядке и как надо делать в каждом случае, и точно известны все случаи и обстоятельства, в которых может оказаться автоматическая система. Другими словами, можно автоматизировать процесс, если он *алгоритмизирован* в математическом понимании слова *алгоритм*.

Естественный предел автоматизации – это непредвиденные условия и невозможность формализации практических действий: это руководство человеческими коллективами, проектирование и эксплуатация крупных технических комплексов, воздействие на живые организмы (на человека), на природу, то есть те случаи, когда приходится взаимодействовать со сложными системами. Точное определение сложных систем тоже будет дано позднее, а пока можно привести только один признак – это неформализуемость ряда процессов, происходящих в системе.

Повышение эффективности такого взаимодействия – это тоже необходимость, и, естественно, есть свои пути решения возникающих тут проблем. Такой путь – это третий уровень системности практической деятельности человека: *кибернетизация*.

Основная идея решения этой проблемы состоит в том, чтобы в случаях, когда автоматизация невозможна, использовать то свойство человека, которое называется *интеллектом*, а именно: способность ориентироваться в незнакомых условиях и находить решения слабо формализованных или неформализованных задач. Это выполнение таких операций, как *экспертная оценка* или сравнение неколичественных или многомерных вариантов, принятие управленческих решений, взятие на себя ответственности и т.п.

Таким образом, получают автоматизированные (в отличие от автоматических) системы. И это не конец кибернетизации, а скорее, ее начало. Дальше – создание искусственного интеллекта. Это отдельный разговор и предмет других курсов, например курса "Прикладные системы искусственного интеллекта".

1.3. Системность познавательных процессов и природы

Сам процесс познания является системным, и знания, добытые человечеством, также системны. Природа бесконечна в своем многообразии. Неограниченно и желание человека в познании окружающего мира. Однако ресурсы человека, как временные, так и материальные, ограничены. Одна из особенностей познания, которая позволяет постепенно, шаг за шагом разрешать это противоречие – это наличие аналитического и систематического образа мышления. Суть анализа – разделение целого на части, представление сложного в виде совокупности простых компонент. Обратный процесс – синтез.

Сами полученные знания тоже, с одной стороны, являются аналитическими, и это отражается в существовании различных узких наук, а с другой – имеется необходимость в синтетических пограничных науках, типа физикохимии, биохимии и т.п. Другая форма синтетических знаний – это существование наук, изучающих общие закономерности, таких, как

философия, метаматематика, а также системных наук – кибернетики, теории систем и других.

По современным представлениям системность понимается не только как свойство человеческой практики, но и как свойство всей материи. Системность мышления вытекает из системности окружающего нас мира. Можно говорить о мире как о бесконечной иерархической системе систем, находящихся на разных стадиях развития, на разных уровнях иерархии, взаимодействующих друг с другом. Таким образом, системность природы не только логически выводится в рамках теоретических построений, но и практически выявляется в реально наблюдаемых явлениях, как с участием человека, так и без него.

1.4. Общие свойства систем

Независимо от того, как понимается системность (системность мышления, познания либо как свойство природы), все исследователи признают, что имеются определенные признаки, свойства, черты, присущие любой системе независимо от ее происхождения.

Приводимый ниже список таких свойств не является, конечно, полным, но наиболее важные моменты в этом списке отражены.

1. Всякая система обладает целостностью, обособленностью от окружающей среды, выступает как нечто отдельное, целое.

2. Обособленность системы не означает ее изолированности: имеется связь с внешней средой, взаимодействие, обмен энергией, материей, информацией. В этом смысле любая система является открытой, незамкнутой.

3. Целостность системы не есть однородность и неделимость: в системе можно выделить определенные составные части.

4. Наличие составных частей не означает, что эти части изолированы друг от друга. Части как раз и образуют систему благодаря связям между ними. Открытость системы (п.2) означает, что ее части связаны и с внешней средой. Целостность же системы (п.1) означает, что внутренние связи между частями, образующими систему, в каком-то смысле важнее, сильнее, чем внешние связи.

5. Целостность системы обусловлена тем, что система обладает такими свойствами, которые отсутствуют у составляющих ее частей. То есть свойства системы не сводятся к свойствам ее частей, не являются простой суммой этих свойств. При объединении частей в систему возникают качественно новые свойства, которые и позволяют выделять и описывать объект именно как систему. Такое возникновение нового качества называется *эмерджентностью* (от английского слова emergence – возникновение из ничего, внезапное появление, неожиданная случайность). Это понятие помогает прояснить разницу между внутренними и внешними связями: свойства системы как целого проявляются в ее взаимодействии с окружающей средой (то есть реализуются через внешние связи как функция системы), однако сами эти свойства возникают и могут существовать только благодаря взаимодействию частей (то есть благодаря внутренним связям, обусловленным структурой системы).

6. Еще один аспект целостности системы: изъятие какой-либо части из системы приводит к потере некоторых существенных свойств системы – получается уже другая система. Изъятая часть также теряет определенные свойства, которые могли реализовываться лишь до тех пор, пока эта часть находилась в системе.

7. Свойство под номером 2, то есть взаимосвязанность системы с окружающей средой, означает, что эта система входит как составная часть в некоторую большую систему. В результате мир можно представить как иерархическую систему вложенных друг в друга, перекрывающихся пол-

ностью или частично либо вообще разделенных, но взаимосвязанных и взаимодействующих систем.

8. При характеристике систем важным моментом является понятие *цели*, которая, в общем, определяет и структуру, и функции системы. Функцию системы можно интерпретировать как проявление целеустремленности системы, то есть функция – это способ достижения системой цели, а структура обеспечивает реализацию этого способа. Рассмотрение целей системы становится одной из важнейших проблем системологии.

9. Важным свойством систем является их *динамика*, то есть изменение во времени в результате внутренних и внешних воздействий. Многие явления в системах невозможно понять без учета их динамики.

2. МОДЕЛИ И МОДЕЛИРОВАНИЕ

2.1. Понятие модели и его развитие

Понятие модели, моделирования, то есть построения, использования и совершенствования моделей чрезвычайно важны в теории систем.

Сначала моделью называли некоторое вспомогательное средство, объект, который в определенной степени заменяет другой объект. Не сразу была понята и всеобщность моделирования: именно не просто *возможность*, но и *необходимость* представления наших знаний в виде моделей. Древние философы, например, считали, что нельзя моделировать естественные процессы, а отображать природу можно только с помощью рассуждений, споров и т.п., то есть с помощью *языковых моделей*, как сказали бы мы сейчас. Таким образом, очень долго понятие «модель» относилось к материальным объектам специального вида: манекен, модели судов, самолетов, чучела.

Осмысление особенностей таких моделей привело к разработке многочисленных определений понятия модели, например: моделью называется объект-заменитель, который в определенных условиях может заменить объект-оригинал, воспроизводя интересующие нас свойства и характеристики оригинала, причем имеет существенные преимущества в виде наглядности, обозримости, доступности испытаний или другие.

Затем были осознаны модельные свойства чертежей, рисунков, планов, карт, то есть объектов искусственного происхождения, реализующих абстракцию довольно высокого уровня.

Очередной этап понимания модели – это признание того, что моделями могут быть не только реальные объекты, но и идеальные абстрактные построения. Классический и наиболее широко применяемый пример таких моделей – *математические модели*. Как известно, моделью в математике

называется алгебраическая структура, в которой множество операций пусто и есть только отношения, то есть модель понимается как результат отображения с помощью этих отношений одного множества математических объектов на другое множество.

В результате развития метаматематики была создана содержательная *теория моделей*.

В 20-м веке понятие модели становится все более широким, включающим и реальные, и идеальные модели. При этом понятие абстрактной модели вышло далеко за пределы математических моделей и стало относиться к любым знаниям и представлениям о мире. Споры, кстати, вокруг такого широкого толкования модели продолжаются и поныне. Например, стоит ли моделями называть такие формы научных знаний, как законы, теории, гипотезы? Отрицательный ответ исходит из исторической, а следовательно, и логической первичности последних понятий, и модель — лишь вспомогательное средство. Положительный ответ основан на признании иерархии моделей, в которой модель более высокого уровня (например теория) содержит модели нижних уровней (гипотезы). Важно и то, что признание перечисленных понятий моделями подчеркивает их относительную истинность.

С философской точки зрения возникает и такой вопрос: не означает ли широкое толкование модели, что это понятие применимо ко всему и, следовательно, является логически пустым? По крайней мере, три момента позволяют отрицательно ответить на этот вопрос. Во-первых, применительно к разным объектам понятие модели имеет и разное содержание (опять вспоминаем иерархию моделей). Во-вторых, из того, что любой объект может быть использован как модель, не следует, что он ничем иным и не является. В-третьих, даже самые общие понятия не являются логически пустыми (материя, система и др.).

Окончательно можно сказать, что сначала в сфере научных дисциплин, таких, как информатика, математика, кибернетика, а затем и в других областях понятие модели стало осознаться как нечто универсальное, хотя и реализуемое различными способами. Можно также сказать, что модель есть способ существования знания.

Важно подчеркнуть также целевой характер модели. Всякий процесс труда (и отдыха, кстати) есть деятельность, направленная на достижение определенной цели. Цель – это образ желаемого будущего, то есть модель состояния, на реализацию которого направлена деятельность. Более того, системность деятельности проявляется помимо прочего и в том, что осуществляется по определенному плану, шаг за шагом. Следовательно, этот план – образ будущей деятельности, ее модель.

Модель, таким образом, не просто объект-заменитель, не вообще какое-то отображение оригинала, а отображение целевое. Модель отображает не сам по себе объект, а то, что в нем нас интересует в соответствии с поставленной целью. Например, лежит камень на дороге. Для проходящего или проезжающего путешественника могут представлять интерес разные модели этого камня, отражающие разные его свойства, в зависимости от целей, которые путешественник перед собой ставит:

- камень как орудие для забивания гвоздя в подметку;
- камень как носитель руды для геолога;
- камень как место для отдыха;
- камень как помеха автомобилю;
- камень как орудие преступления для бандита с большой дороги;
- и т.д.

2.2. Типы моделей

Поскольку модель является целевым отображением, то и различных моделей одного и того же объекта может быть множество. Целевая предназначенность моделей позволяет все множество моделей разделить на основные типы – по типам целей.

2.2.1. Модели прагматические и познавательные

Один из примеров такого деления целей – это цели теоретические и практические. В соответствии с этим можно говорить о моделях *познавательных и прагматических* соответственно. В известной степени это деление условно, но и различия достаточно очевидны. Основное отличие между этими типами моделей – в отношении к оригиналу.

Познавательные модели являются формой организации и представления знаний, средством для соединения новых знаний с уже имеющимися. Поэтому в случае расхождения между моделью и реальностью эти расхождения устраняются путем изменения модели.

Прагматические модели – это средство управления, средство организации практического действия, образцово-правильные действия или их результаты. Поэтому использование прагматических целей предполагает изменение реальности с тем, чтобы приблизить реальность к модели.

Например, карта или план местности может выступать как познавательная модель для картографа, составляющего или уточняющего эту карту, и может являться прагматической моделью для строителя, осуществляющего возведение каких-либо объектов по данному плану.

2.2.2. Модели статические и динамические

Еще один достаточно важный вид классификации моделей - модели статические и динамические. Когда нас в конкретном объекте интересует не изменение его состояния во времени, а некоторое фиксированное, статическое состояние, мы имеем дело со *статическими* моделями. Структурная модель системы – яркий пример статической модели. Если же наши цели связаны с различием между состояниями, с их изменениями, с их динамикой, то возникает необходимость в отображении процесса изменения состояний. И такие модели называют *динамическими*.

2.2.3. Абстрактные модели

По способам реализации, воплощения моделей их также можно классифицировать. Нас пока интересуют модели, создаваемые человеком, а в его распоряжении есть два типа материала для построения моделей - средства самого сознания и средства окружающего материального мира. А раз так, то и модели можно поделить на *абстрактные* (идеальные) и *материальные* (реальные или вещественные).

Абстрактные модели – это идеальные конструкции, построенные средствами сознания, мышления. Особый интерес из них представляют модели, предназначенные для общения между людьми, а среди этих моделей - модели, создаваемые средствами языка, то есть языковые модели.

Естественный язык (русский, английский, немецкий и др.) является универсальным средством построения любых абстрактных моделей. Эта универсальность обеспечивается не только возможностью введения новых слов в язык, но и возможностью иерархического построения все более развитых языковых моделей (слово–предложение–текст). Неоднозначность, размытость, неопределенность естественного языка (начиная уже на уров-

не слов) также способствует универсальности языковых моделей. На практике эта неоднозначность ликвидируется с помощью "понимания" и "интерпретации". Но рано или поздно эта размытость начинает мешать, и тогда на помощь приходят профессиональные языки, вырабатываемые людьми одной профессии.

Наиболее ярко это проявляется в конкретных науках. Дифференциация наук потребовала создания специализированных языков, более точных, емких и конкретных, чем естественный язык. Языковые модели специальных наук более точны, более компактны, содержат больше информации.

Для представления новых знаний требуются новые модели, и если существующих языковых средств не хватает для построения этих моделей, то возникают еще более специализированные языки. В результате приходим к иерархии языков и соответствующей иерархии моделей (см. рис 2.1). На нижнем уровне каждой ветви этого иерархического дерева расположены модели, имеющие максимально достижимую точность и определенность для сегодняшнего состояния области знаний. В идеале на нижнем уровне находятся математические модели, созданные на языке математических символов и обладающие абсолютной точностью.

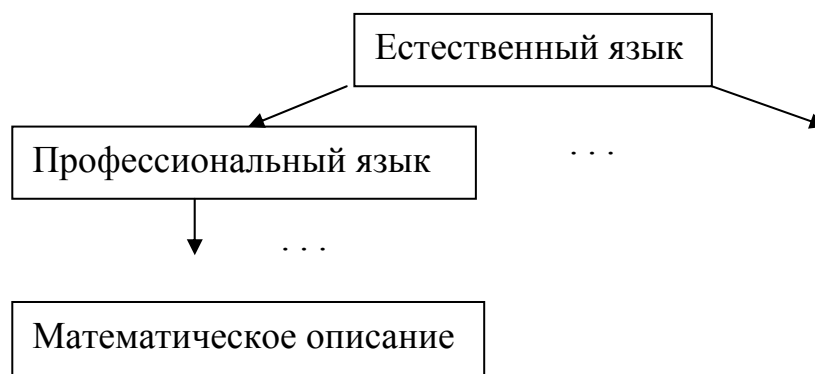


Рис. 2.1. Иерархия языковых моделей

2.2.4. Материальные модели

Чтобы материальная конструкция могла быть отображением некоторого объекта, то есть замещала в каком-то отношении оригинал, между моделью и оригиналом должно быть установлено отношение подобия, похожести. Можно выделить три основных способа установления такого подобия.

1. Подобие, устанавливаемое в результате физического взаимодействия в процессе создания модели. Это *прямое* подобие. К таким моделям относятся масштабированные модели кораблей, самолетов, макеты зданий, протезы, шаблоны, выкройки, фотографии. Прямое подобие может обладать лишь отдаленным сходством с оригиналом, но только при прямом подобии возможна взаимозаменяемость оригинала и модели. Но все же, как бы ни была хороша модель, возникают проблемы переноса результатов моделирования на оригинал. Такие проблемы часто становятся нетривиальными. В связи с этим и возникла разветвленная, содержательная *теория подобия*, относящаяся именно к моделям прямого подобия.

2. Второй тип подобия в противоположность первому назовем *косвенным*. Оно устанавливается не в результате физического взаимодействия оригинала и модели, а объективно существует в природе и обнаруживается в виде совпадения (или достаточной близости) их абстрактных моделей. Наиболее яркий пример - электромеханическая аналогия. Механические и электрические процессы описываются одними и теми же уравнениями (т.е. имеют одинаковые абстрактные модели), различие состоит лишь в разной физической интерпретации входящих в эти уравнения переменных. В результате эксперименты с громоздкой и неудобной механической системой могут быть заменены более удобным манипулированием с электрической схемой. Роль аналогий, то есть косвенного подобия, в науке, технике и практике трудно переоценить, они часто незаменимы.

3. Третий тип реальных моделей образуют модели, в которых подобие оригиналу устанавливается в результате некоторого соглашения, договоренности. Назовем это подобие *условным*. Пример: деньги (модель стоимости), сигналы (модель сообщений), рабочие чертежи (модели будущей конструкции). Модели условного подобия являются, по сути, способом материальной реализации абстрактных моделей, вещественной формой, в которой абстрактные модели воспринимаются, хранятся и передаются от человека к человеку. Достигается это соглашением о том, какое состояние оригинала ставится в соответствие данному элементу абстрактной модели. Такое соглашение принимает вид набора правил для построения моделей условного подобия и правил пользования ими.

В конкретных науках эта общая схема уточняется и наполняется содержанием. Ряд наук имеют дело со специфическими моделями условного подобия, которые применяются в технических системах без участия человека. Это теория связи, теория информации, радиотехника, теория управления и другие. С других позиций рассматриваются модели условного подобия там, где эти модели используются самим человеком. При этом для использования таких моделей применяются специальные средства — *знаки*. Знаками могут быть буквы алфавита, математические значки, звуки, движения брачного танца у животных и т.п. Наука об осмысленных знаковых системах называется *семиотикой*. Последовательности знаков — это тексты. Правила, определяющие множество текстов, называются *синтаксисом*. Описание множества смыслов и его соответствие множеству текстов образует *семантику*.

2.3. Свойства моделей

2.3.1. Условия реализации моделей

Чтобы модель отвечала своему назначению, недостаточно иметь модель саму по себе: необходимы и определенные условия, обеспечивающие ее функционирование. Отсутствие или недостаточность таких условий лишает модель ее модельных свойств, не позволяет раскрыть ее потенциальные возможности. Необходимо, чтобы модель была согласована с окружающей средой, в которой ей предстоит функционировать. Это самое общее свойство моделей при необходимости можно и конкретизировать, выявляя отдельные аспекты такого согласования. В частности, очень важным моментом является обеспечение ресурсами (в том числе и материальными). Кроме того, не только в модели должны быть интерфейсы со средой, но и в самой среде должны быть реализованы подсистемы, другие модели и алгоритмы, потребляющие результаты ее функционирования, управляющие и контролирующие ход процесса моделирования.

2.3.2. Различия между моделью и оригиналом

Рассмотрим те свойства модели, которые определяют ценность самого моделирования, то есть отношение моделей с отображаемым ими объектами: чем отличаются модели от оригинала и что у них общего. Главные отличия модели от оригинала это конечность, упрощенность и приближенность.

Конечность абстрактных моделей сомнений не вызывает, так как они сразу наделяются конечным набором свойств. Но модели материальные – это некоторые вещественные объекты и как всякие объекты они бесконечны, в том числе и в своих связях с другими объектами. Здесь-то и

проявляется различие между самим объектом и тем же объектом, используемым в качестве модели другого объекта. Из необозримого множества свойств объекта – модели выбираются, рассматриваются и используются только некоторые свойства, подобные интересующим нас свойствам объекта-оригинала. Наиболее наглядно конечность видна в знаковых моделях. Таким образом, модель подобна оригиналу в конечном числе отношений – это главный аспект конечности реальных моделей.

Следующие факторы позволяют с помощью конечных моделей отображать бесконечную действительность (и не просто отображать, а отображать эффективно, то есть достаточно правильно): упрощенность и приближенность модели.

Можно, прежде всего, отметить, что сама конечность моделей делает их упрощенность неизбежной, но это ограничение не настолько сильно, как кажется на первый взгляд. Гораздо важнее то, что в человеческой практике упрощенность является вполне допустимой, а для некоторых целей не только достаточной, но и необходимой.

Какие из свойств объекта включать в модель, а какие нет, зависит от целей моделирования, и выбор цели определит, что можно и нужно отбросить и в каком направлении упрощать модель. Упрощение – сильное средство в выявлении главных эффектов; идеальный газ, несжимаемая жидкость, абсолютно черное тело и т.д.

Следующая причина упрощенности модели – необходимость оперировать с ней и связанное с этим ресурсное ограничение. Мы вынужденно упрощаем модель, так как не знаем, как работать со сложной моделью или у нас нет требуемых ресурсов (материальных, энергетических, временных) для создания сложной модели.

Под *приближенностью* (приблизительностью) отображения действительности с помощью модели будем иметь в виду различия, описывае-

мые отношением порядка: количественные (больше – меньше) или хотя бы ранговые (лучше – хуже).

Приближенность модели может быть очень высокой (удачные подделки, например, денег), а может быть видна сразу или варьироваться (географическая карта в разных масштабах), но во всех случаях модель – это другой объект и различия неизбежны. Мету различий мы можем ввести, только соотнеся эти различия с целью моделирования (опять цель!).

2.3.3. Сходство между моделью и оригиналом

Общность модели и моделируемого объекта можно пояснить понятием *адекватности*. Модель, с помощью которой успешно достигается поставленная цель, назовем адекватной этой цели. Такое определение адекватности, вообще говоря, не полностью совпадает с полнотой, точностью и истинностью модели, а лишь в той мере, которая необходима для достижения цели моделирования. В некоторых случаях удастся ввести меру адекватности модели, то есть указать способ сравнения двух моделей по степени успешности достижения цели. Если такой способ приводит еще и к количественной оценке адекватности, то задача улучшения модели существенно облегчается. В этих случаях можно количественно ставить (и успешно решать!) следующие задачи:

- задачи по *идентификации* модели (то есть о нахождении в заданном классе моделей наиболее адекватной),
- задачи по исследованию *чувствительности и устойчивости* модели (то есть зависимости меры адекватности модели от ее точности),
- задачи по *адаптации* модели (то есть настройки параметров или структуры модели с целью повышения меры адекватности) и т.д.

Теперь еще раз вернемся к истинности модели. Поскольку различия между моделью и реальностью неизбежны и неустранимы, возникает во-

прос: существует ли предел истинности, правильности наших знаний, сконцентрированных в моделях? Рассмотрение проблемы истинности знаний с философских позиций оставим философам, а наша задача конкретнее: обратить внимание на сочетание истинного и предполагаемого (могущего быть как истинным, так и ложным) во всех моделях.

Об истинности и ложности модели самой по себе говорить бессмысленно: только практическое соотнесение модели с отображаемым оригиналом выявляет степень истинности. При этом изменение условий, в которых ведется сравнение, весьма существенно влияет на результат. Каждая модель явно или неявно содержит условие своей истинности, и одна из опасностей (и весьма существенная) практики моделирования состоит в применении модели без проверки выполнения этих условий.

Еще один важный момент соотношения истинного и предполагаемого при построении моделей состоит в том, что ошибки в предположениях имеют разные последствия для прагматических и познавательных целей. Если для первых они нежелательны и даже вредны, то для вторых поисковые предположения, гипотезы, истинность которых еще предстоит проверить – единственная возможность оторваться от фактов.

Весьма важным свойством любой модели является динамика. Как и все в мире, модели проходят свой жизненный цикл: они возникают, развиваются, сотрудничают или соперничают с другими моделями, прекращают свое существование. Изучать динамику развития модели невозможно без моделирования самого процесса моделирования, отдельных его этапов, шагов, последовательностей действий. Многие исследователи искали последовательность наиболее эффективных этапов при работе с моделью, пытались алгоритмизировать этот процесс. Но выяснилось, что не существует единого, пригодного для всех случаев алгоритма работы с моделью. Этому можно привести много причин, но с методологической точки зре-

ния – это одна из алгоритмически неразрешимых проблем в рамках теории алгоритмов.

Резюмируя вышесказанное, можно дать одно из многочисленных определений модели. Модель есть отображение: целевое, абстрактное или реальное, статическое или динамическое, согласованное со средой, конечное, упрощенное, приближенное, имеющее наряду с безусловно-истинным условно-истинное и ложное содержание, проявляющееся и развивающееся в процессе его создания и использования.

Если кому-то это определение покажется слишком длинным и утомительным, тот вполне может заменить его более коротким: **модель есть системное отображение действительности.**

3. СИСТЕМЫ, ИХ ОБЩЕЕ ОПИСАНИЕ И КЛАССИФИКАЦИЯ

Перейдем теперь к системам. Понятие системы очень важно, и многие авторы анализировали это понятие, уточняли и развивали его до разной степени формализации. Существует очень много определений системы. Вспоминая то, что говорилось о моделях, эта множественность понятна – определение есть не что иное как модель (языковая) и, следовательно, различие целей и требований к модели приводит к разным определениям.

Попытаемся дать определение системы в развитии, поэтапно уточняя, развивая и конкретизируя модель на пути от естественно-языковой до математической.

3.1. Первое определение системы. Модель "Чёрный ящик"

Рассмотрим вначале искусственные системы. Уже говорилось о том, что человеческая деятельность целенаправленна. Наиболее ярко это проявляется в трудовой деятельности. Однако цель, которую человек перед собой ставит, редко достигается только собственными возможностями. Такое несоответствие желаемого и действительного можно охарактеризовать как *проблемную ситуацию*. Проблемная ситуация развивается постепенно: от неосознанного чувства неудовлетворенности к осознанию потребности, к выявлению проблемы и далее – к формулировке цели. Последующая деятельность направлена на достижение этой цели. Укрупненно, в общих чертах ее можно описать как отбор из окружающей среды объектов, свойства которых можно использовать для достижения цели и на объединение этих объектов надлежащим образом, то есть как работу по созданию

того, что, собственно, и называется *системой*. Это и есть первое определение системы: **система – это средство достижения цели**.

Сформулировать цель порой бывает непросто. Одна из причин этого та, что между целью (то есть абстрактной и конечной моделью) и реальной системой нет и не может быть однозначного соответствия: для достижения данной цели могут использоваться разные средства (системы), а заданную реальную систему можно использовать и для других целей, прямо не предусмотренных при ее создании.

Четкая формулировка, постановка цели в инженерной практике – один из важнейших этапов создания систем. Обычно этот этап идет итерационно, с постепенным уточнением и конкретизацией целей.

Первое определение системы не только отвечает на вопрос: “Зачем нужна система?”, но и конструктивно указывает, какие объекты следует, а какие не следует из окружающей среды включать в систему: включаются такие объекты, свойства которых во взаимосвязи с уже включенными объектами позволяют достигать поставленную цель.

В первом определении системы акцент сделан на назначение системы и об ее устройстве почти ничего не говорится. Человеку удобнее работать с наглядными, образными моделями, поэтому представим языковую модель первого определения системы в виде визуального эквивалента.

О внутреннем устройстве системы ничего не известно, поэтому изобразим систему в виде ящика с непрозрачными (чёрными) стенками. Уже при таком изображении модель отражает два свойства системы: целостность и обособленность от среды.

Далее в определении хоть и косвенно, но сказано, что система не полностью изолирована. Достигнутая цель – это запланированные изменения в окружающей среде, некоторые продукты работы системы, предназначенные для потребления вне данной системы. Система связана со сре-

дой и с помощью этих связей воздействует на среду. Изобразим эти связи стрелками, выходящими из системы. Это выходы системы.

Наконец, в определении есть намек и на связи другого типа: система является средством, поэтому должны существовать и возможности ее использования, воздействия на нее, то есть такие связи, которые направлены из среды в систему. Это входы системы.

В результате получим модель системы, которая называется *чёрным ящиком* (см. рис. 3.1). Эта модель, несмотря на внешнюю простоту, часто оказывается полезной.

В некоторых случаях достаточно содержательного описания входов и выходов системы, тогда модель чёрного ящика – это их список. В других случаях требуется количественное описание некоторых или всех входов и выходов. Чтобы максимально формализовать модель “чёрного ящика”, необходимо задать два множества X и Y входных и выходных переменных.

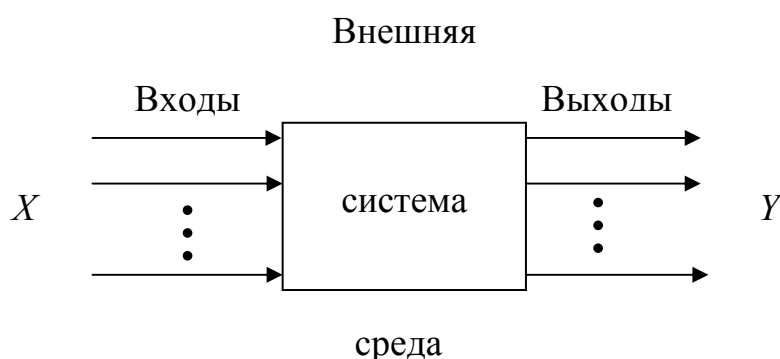


Рис.3.1. Модель системы в виде «чёрного ящика»

Задание этих множеств – сама по себе задача не тривиальная даже для конкретной системы, так как на вопрос о том, какие и сколько связей следует включать в модель – ответ не прост и не однозначен. Дело в том, что любая реальная система, как и любой объект, взаимодействует с объектами окружающей среды неограниченным числом способов. Выбрать для построения модели из этого бесконечного множества связей конечное их множество – задача часто непростая. Критерий отбора здесь – целевое

назначение системы, существенность той или иной связи по отношению к цели. А вот при оценке этой существенности и могут возникать ошибки.

Особенно важно учитывать этот момент при задании цели системы, то есть при определении выходов. При этом основную цель приходится сопровождать заданием *дополнительных* целей, невыполнение которых может сделать ненужной или даже вредной достижение основной цели.

Модель “чёрного ящика” оказывается часто не только полезной, но и единственно возможной при изучении систем. Например, это бывает, когда речь идет об исследовании живых организмов в естественных для них условиях, где следует специально заботиться о том, чтобы измерения как можно меньше влияли на систему.

3.2. Модель состава системы

Вопросы внутреннего устройства системы на уровне модели “чёрного ящика” остаются открытыми. Для этого нужны более детальные, более развитые модели. Такие свойства системы, как целостность и обособленность, отображенные в модели “чёрного ящика”, выступают как внешние свойства. Внимательное рассмотрение системы говорит о том, что внутренность “чёрного ящика” оказывается неоднородной. Это позволяет различать составные части системы, некоторые из которых при еще более детальном рассмотрении, в свою очередь, могут быть разбиты на составные части и т.д. Части, которые мы считаем неделимыми, называются *элементами*. Части, состоящие более чем из одного элемента, называются *подсистемами*. В результате получается модель состава системы, описывающая из каких подсистем и элементов она состоит.

Построение модели состава системы — дело не простое и не однозначное. Можно перечислить ряд причин этого.

Понятие элементарности можно определять по-разному. То, что с одной точки зрения является элементарным, с другой – можно представить как подсистему, требующую дальнейшего деления.

Как и любая модель, модель состава является целевой, и для разных целей может быть разбита различным образом.

Модели состава одной и той же системы различаются потому, что всякое деление целого на части в достаточной степени условно. Границы между подсистемами условны, относительны. Это же относится и к границам самой системы с окружающей средой.

Таким образом, модель состава ограничена снизу тем, что считается элементом, и сверху – границей системы. Как эта граница, так и границы разбиения на подсистемы определяются целями построения модели и, следовательно, не абсолютны.

3.3. Модель структуры системы. Второе определение системы

Для определенных задач вполне достаточно иметь модель системы в виде “черного ящика” или модели состава. Но ясно, что есть вопросы, которые нельзя решить только на уровне этих моделей. Необходимо еще соединить правильно элементы и подсистемы, то есть установить между составными частями системы определенные связи. Совокупность необходимых и достаточных для достижения цели отношений между элементами называется *структурой системы*.

Перечень связей между элементами (структура системы) является отвлеченной, абстрактной моделью: установлены только отношения между элементами, но ничего не говорится о самих элементах. На практике обычно сначала описываются сами элементы (модель структуры), но теоретически можно исследовать отдельно модель структуры. Опять-таки, из

множества реальных отношений между элементами в модель структуры включаются только те, которые важны, существенны для достижения цели.

Суммируя, объединяя все три рассмотренные модели, можно сформулировать второе определение системы: **это совокупность взаимосвязанных элементов, обособленная от среды и взаимодействующая с ней как целое.** Графическое изображение такой модели, объединяющей и модель “черного ящика” и модель состава и модель структуры, называется *структурной схемой системы*.

На структурной схеме указаны все элементы, все связи между элементами и связи определенных элементов с окружающей средой (входы и выходы системы). Все структурные схемы имеют нечто общее и, если абстрагироваться от содержательной стороны структурных схем, в результате получим элементы и связи между ними, а также (если это необходимо) пометки для различения элементов и (или) связей. Это есть не что иное, как ориентированный (возможно, взвешенный) граф, и эффективное изучение структурных схем осуществляется с помощью *теории графов*.

Для ряда исследований одной структурной информации, содержащейся в графе, оказывается недостаточно, и тогда методы теории графов становятся вспомогательными, а главным является рассмотрение конкретных функциональных связей между входными, внутренними и выходными переменными систем.

3.4. Динамические модели системы

Все рассмотренные выше модели отображали систему в некоторый фиксированный момент времени, были как бы застывшими снимками системы. В этом смысле их можно назвать *статическими*, подчеркивая их неподвижный, застывший, неизменный характер. Следующий шаг – это понять и описать, как система работает, что происходит с ней самой и с ок-

ружающей средой во времени, в ходе реализации поставленной цели. И подход, и описание, и степень подробности этого описания могут быть различными, но общее у всех этих моделей – это то, что они должны отражать поведение систем, описывать происходящее во времени изменения, последовательности этапов, операций, действий.

Системы, в которых происходят изменения со временем, называются *динамическими*, а соответствующие модели, отображающие это изменение – *динамическими моделями*.

3.4.1. Типы динамических моделей

Для разных систем и объектов разработано большое число динамических моделей, описывающих процессы с разной степенью подробности: от общего понятия динамики, движения вообще – до математических моделей конкретных процессов. Развитие моделей при этом происходит примерно также: от “чёрного ящика” к структурной схеме.

Уже на уровне “черного ящика” различают два типа динамики систем – это *функционирование и развитие*. Под функционированием понимают процессы, которые происходят в системе (и среде), стабильно реализующей фиксированную цель. Развитием называют то, что происходит с системой при изменении ее целей, когда существующая структура перестает удовлетворять новую цель. Не следует считать, что система только либо функционирует, либо развивается. Могут быть разные соотношения между ее подсистемами.

В следующих уровнях динамических моделей происходит уточнение, конкретизация происходящих процессов, различаются части, этапы процесса, рассматриваются их взаимодействия. То есть типы динамических моделей те же, что и статических, только элементы этих моделей имеют временной характер.

Динамический вариант “чёрного ящика” – это начальное состояние системы (вход “чёрного ящика”) и конечное (выход). Модель состава – перечень этапов. Структурная схема (или “белый ящик”) – подробное описание происходящего или планируемого процесса.

Те же типы моделей прослеживаются и при более глубокой формализации динамических моделей. Например, при математическом моделировании некоторого процесса его конкретная реализация описывается в виде соответствия между элементами множества X возможных значений $x \in X$ и элементами упорядоченного множества T моментов времени $t \in T$, то есть в виде отображения $T \rightarrow X$. С помощью этих понятий можно строить математические модели систем.

Если рассматривать выход системы $y(t)$ (в общем случае вектор) как ее реакцию на управляемые $U(t)$ и неуправляемые $V(t)$ входы $x(t) \in \{U(t), V(t)\}$, то можно модель “черного ящика” изобразить как совокупность двух процессов: $X^T = \{x(t)\}$ и $Y^T = \{y(t)\}$, $t \in T$ (см. рис. 3.2).

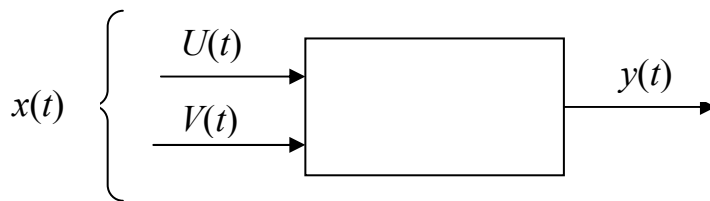


Рис.3.2. Динамическая модель «черного ящика»

Даже если считать $y(t)$ результатом некоторого преобразования Φ процесса $x(t)$: $y(t) = \Phi(x(t))$, то модель “чёрного ящика” предполагает, что это преобразование неизвестно. В случае перехода к “белому ящику” это соответствие между входом и выходом можно описать тем или иным способом. Какой именно применить способ зависит от того, что нам известно о системе и в какой форме эти знания можно использовать.

3.4.2. Общая математическая модель динамической системы

Сложность построения моделей заключается в том, что в общем случае выход системы определяется не только значением входа в данный момент, но и предыдущими значениями входа. Кроме того, в самой системе с течением времени как под действием входных процессов, так и независимо от них могут происходить изменения. Все это требует отражения в модели. В наиболее общей модели это обеспечивается введением понятия *состояние системы*, как некоторой внутренней характеристики системы.

Входные величины в качестве причины определяют изменения во времени всех переменных системы и, в частности, всех выходных величин. Значения этих величин в определенный, заданный момент времени t в общем случае зависит от изменений во времени входных величин на интервале $(-\infty, t]$, то есть значение выхода определяется, как правило, всей предысторией изменения входа. В случае, если предшествующая данному моменту эволюция входных величин известна не полностью, а только, скажем, на интервале $[t_0, t]$ ($t_0 < t$), предшествующем моменту времени t , то может оказаться, что в общем случае этой информации будет недостаточно для определения внутренних переменных системы и выходных величин в текущий момент времени. Однако в том случае, когда имеется дополнительная информация о значениях определенных переменных системы в момент времени t_0 , значения выхода снова могут быть определены полностью, так же как и значения всех внутренних переменных. Таким образом, отсутствие информации о динамике входных величин на интервале $(-\infty, t_0)$ можно компенсировать тем, что известны значения некоторых переменных системы в момент времени t_0 . Такие переменные называются *переменными состояниями системы*.

Если обозначить переменные состояния через $\mathbf{q}(t)$ (в общем случае это вектор размерностью n), входные переменные через $\mathbf{x}(t)$ (размерность вектора $\mathbf{x}(t)$, т.е. число входов – m), а выходные переменные через $\mathbf{y}(t)$ (вектор размерностью k) (см. рис. 3.3), то соответствующие отображения можно записать как

$$\begin{aligned} [\mathbf{q}(t_0), \mathbf{x}(\tau)] &\rightarrow \mathbf{q}(t); \\ [\mathbf{q}(t_0), \mathbf{x}(\tau)] &\rightarrow \mathbf{y}(t) \end{aligned} \quad (3.4.1)$$

или, что то же самое,

$$\begin{aligned} \mathbf{q}(t) &= \delta(\mathbf{q}(t_0), \mathbf{x}(\tau)); \\ \mathbf{y}(t) &= \lambda(\mathbf{q}(t_0), \mathbf{x}(\tau)). \end{aligned} \quad (3.4.2)$$

В данной записи следует под обозначением τ понимать не точку на оси времени, а весь интервал от t_0 до t .

Отображение δ часто называют переходным, а отображение λ – отображением выхода.

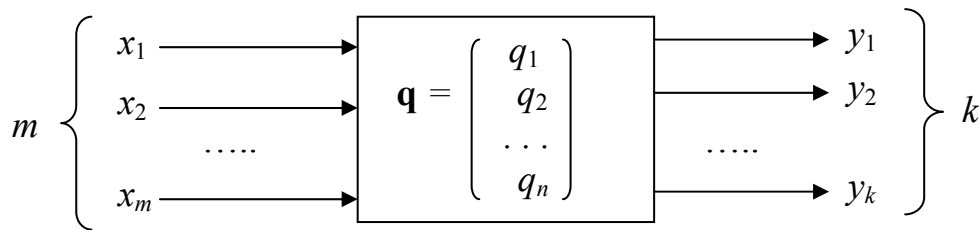


Рис.3.3. Математическая модель динамической системы

Естественно, что векторная запись уравнений (3.4.2) предполагает, что δ и λ также векторы.

Для каждого фиксированного τ значение вектора $\mathbf{q}(\tau)$ задает состояние динамической системы в момент времени τ . Множество Q^n всех

векторов \mathbf{q} образует алфавит состояний динамической системы или *пространство состояний*. Таким образом, $\mathbf{q}(\tau)$ можно трактовать как точку в пространстве состояний.

Пример 3.1. Рассмотрим электрическую цепь (четырёхполюсник) (рис. 3.4).

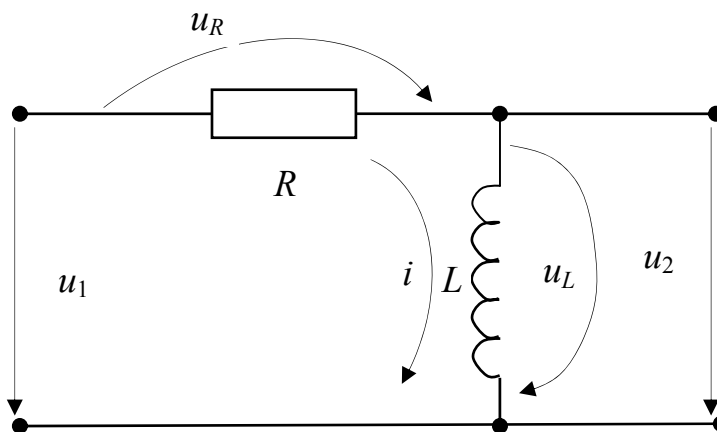


Рис.3.4. Пример динамической системы (электрический четырёхполюсник)

Пять существующих в этой цепи физических величин (четыре напряжения u_1 , u_2 , u_R , u_L и ток i) связаны согласно законам Ома и Кирхгофа, соотношениями

$$\begin{aligned} u_1 &= u_R + u_L, \\ u_R &= R \cdot i, \\ u_2 &= u_L, \\ u_L &= L \cdot \frac{di}{dt}. \end{aligned} \tag{3.4.3}$$

Предполагается, что индуктивность L и сопротивление R не зависят от тока, напряжения и времени, то есть постоянны.

С учетом электротехнических применений данной цепи эти пять физических величин (переменных системы) можно разделить на:

- заданную величину u_1 (причина, вход, входное воздействие);

- величину, предназначенную для некоторых целей u_2 (следствие, выход, выходное воздействие);

- величины, участвующие в преобразовании входного воздействия u_1 в выходное воздействие u_2 : промежуточные величины u_R , u_L , i (внутренние переменные системы).

Решение уравнений (3.4.3) относительно, например, тока i и выхода u_2 для $t \geq t_0 = 0$ будет равно

$$\begin{aligned} i(t) &= \left(i(0) + \frac{1}{L} \int_0^t u_1(\tau) e^{\frac{R}{L}\tau} d\tau \right) \cdot e^{-\frac{R}{L}t}, \\ u_2(t) &= L \frac{di}{dt}. \end{aligned} \quad (3.4.4)$$

Если входная величина $u_1(\tau)$ непрерывна на интервале $[0, t] = I$, то переменная системы $i(t)$ для каждого момента времени $t \in I$ однозначно определяется по значению тока в момент времени $t=0$ и по входной величине $u_1(\tau)$ на интервале $0 \leq \tau < t$. Причем, это происходит независимо от того, принимали различные входные величины или другие переменные системы значения, отличные от нуля при $t < 0$, или нет. Таким образом, воздействие, оказываемое $u_1(t)$ на систему до момента времени $t=0$, например, на изменение $i(t)$ при $t < 0$, можно учесть только в одном значении тока в момент времени $t=0$. Это же самое относится и к выходной величине $u_2(t)$. Приведенные рассуждения позволяют уравнения (3.4.4) записать в символической форме аналогично соотношениям (3.4.1):

$$\begin{aligned} [i(0), u_1(\tau)] &\rightarrow i(t), \\ [i(0), u_1(\tau)] &\rightarrow u_2(t), \\ (0 \leq \tau < t). \end{aligned}$$

Из последних соотношений следует, что ток i является переменной состояния системы, изображенной на рис. 3.4.

Исходя из основных уравнений динамической системы (3.4.2), можно установить, что для всестороннего описания системы должны быть заданы три основные множества:

- а) множество X значений входных величин x (входной алфавит);
- б) множество Y значений выходных величин y (выходной алфавит);
- в) множество Q значений переменных состояния q (алфавит состояний).

Вектор $q(t)$ (также как и вектор $q(\tau)$) является тогда элементом n -кратного декартового произведения Q^n : $q(t) \in Q^n$. Соответственно для k выходных величин вектор $y(t)$ является элементом k -кратного декартового произведения Y^k : $y(t) \in Y^k$.

Вообще говоря, не обязательно эти множества должны быть множеством действительных чисел R , т.е. не обязательно должно быть $X=Y=Q=R$. Чтобы получить более общее определение системы, под множествами X, Y, Q следует понимать множества более общего вида. При этом каждая компонента $x_\nu(\tau)$ ($\nu=1, 2, \dots, m$) входного вектора $x(\tau)$ ($t_0 \leq \tau < t$) уже определяется как некоторое отображение интервала $T_t = [t_0, t]$ действительной временной оси $T=R$ или в более общем случае множества $T_{0t} = T_t \cap T_0$ ($T_0 \subset R$) на множество X :

$$x_\nu(\tau): T_{0t} \rightarrow X, \quad T_{0t} = T_t \cap T_0, \quad T_0 \subset R,$$

$$T_t = (t_0, t), \quad \nu = 1, 2, \dots, m.$$

Вектор $x(\tau) \in X^n$. Если через $\mathcal{X} = \{T_{0t} \rightarrow X^m\}$ обозначить множество всех отображений всех множеств T_{0t} в декартово произведение X^m (или в некоторое подмножество этого множества), то соотношения (3.4.2.) можно записать как отображения вида

$$\begin{aligned} \delta: Q^n \times \mathcal{X} &\rightarrow Q^n; \\ \lambda: Q^n \times \mathcal{X} &\rightarrow Y^k. \end{aligned} \tag{3.4.5}$$

Конкретизируя множества \mathcal{X}, Y и Q , приходим к математическим моделям различных систем. Если эти множества *конечны*, то такая система будет называться *конечным автоматом*, и для ее исследования разработана соответствующая теория конечных автоматов. Это довольно простой класс систем в том смысле, что для исследования конечных автоматов необходимы лишь методы логики, алгебры и теории множеств. И в то же время это достаточно важный и широкий класс систем, так как в него входят все дискретные (цифровые) измерительные, управляющие и вычислительные устройства.

Если \mathcal{X}, Y^k и Q^n являются метрическими или в более общем случае топологическими пространствами, а отображения λ и δ непрерывны в этих пространствах, то мы переходим к так называемым *гладким* системам. Для такого класса систем характерно, что переходное отображение δ является общим решением векторно-матричного уравнения:

$$\frac{d\mathbf{q}}{dt} = \mathbf{f}(t, \mathbf{q}, \mathbf{x}), t \in R. \quad (3.4.6)$$

Если же в уравнении (3.4.6) время t является элементом счетного множества, т.е. течет дискретно, шагами или квантами (это будет в случае, если $T_0 \subseteq N_0$, где N_0 – множество натуральных чисел), то от выражения (3.4.6) приходим к разностному уравнению:

$$\mathbf{q}(t_{k+1}) = \mathbf{f}(t_k, \mathbf{q}, \mathbf{x}), k \in N_0, \quad (3.4.7)$$

являющемуся описанием дискретных во времени системы.

Если отображение λ не зависит явно от времени и отображение δ инвариантно (неизменно) к сдвигу во времени, то получаем описание *стационарных систем*. У таких систем их свойства со временем не меняются.

Особо нужно остановиться на таком свойстве реальных систем, как принцип причинности. Этот принцип означает, что отклик (выход) системы на некоторые воздействия не может появиться раньше самого воздействия. Это условие не всегда выполняется в рамках математических моделей систем, и одна из проблем теории таких систем состоит как раз в выяснении условий физической реализуемости теоретических моделей.

Следует также отметить, что в основных уравнениях (3.4.1) – (3.4.2) всегда предполагается, что $t > t_0$. Физически это означает, что из состояния системы в данный момент времени $\mathbf{q}(t_0)$ можно сделать заключение о поведении системы только в последующие моменты времени. Таким образом, не предполагается, что по заданным значениям $\mathbf{q}(t_0)$ и $\mathbf{x}(\tau)$ можно также определить и $\mathbf{q}(t)$ при $t < t_0$ ($t < \tau \leq t_0$). Иными словами, если известно конечное состояние $\mathbf{q}(t_0)$ и входное воздействие $\mathbf{x}(\tau)$ на предшествующем интервале времени $t < \tau \leq t_0$, то восстановить первоначальное состояние $\mathbf{q}(t)$ не всегда представляется возможным. Эти же рассуждения справедливы и для $\mathbf{y}(t)$. Это означает, что будущее поведение некоторой динамической системы зависит от всей ее предшествующей эволюции только в той мере, насколько эта предыстория влияет на начальное состояние. То есть динамические системы по определению не обязательно являются обратимыми (во времени).

3.5. Классификация систем

Когда мы сравниваем системы, начинаем их различать, мы тем самым вводим некоторую классификацию. *Основания классификации*, т.е. свойства систем, по которым мы их различаем, могут быть самыми различными. Необходимо только помнить, что всякая классификация – это только модель реальности и, как любая модель, конечна, упрощенна, при-

ближенна и условна. Разграничение внутри класса приводит к подклассам и, в конце концов, получается многоуровневая, иерархическая классификация. Важным моментом является полнота классификации. Когда нет уверенности в полноте классификации, имеет смысл вводить класс “все остальное”.

3.5.1. Классификация по происхождению

Системы можно разделить на искусственные (т.е. созданные человеком) и естественные. Для полноты классификации следует выделить еще класс систем, объединяющих искусственные и естественные подсистемы. Двухуровневая классификация систем приведена на рисунке 3.5. Неполнота классификации на втором уровне для всех трех классов очевидна. Примером подклассов смешанных систем могут служить эргономические системы (человек - оператор и машина), биотехнические системы (системы, в которые входят живые организмы и технические устройства), автоматизированные системы управления.

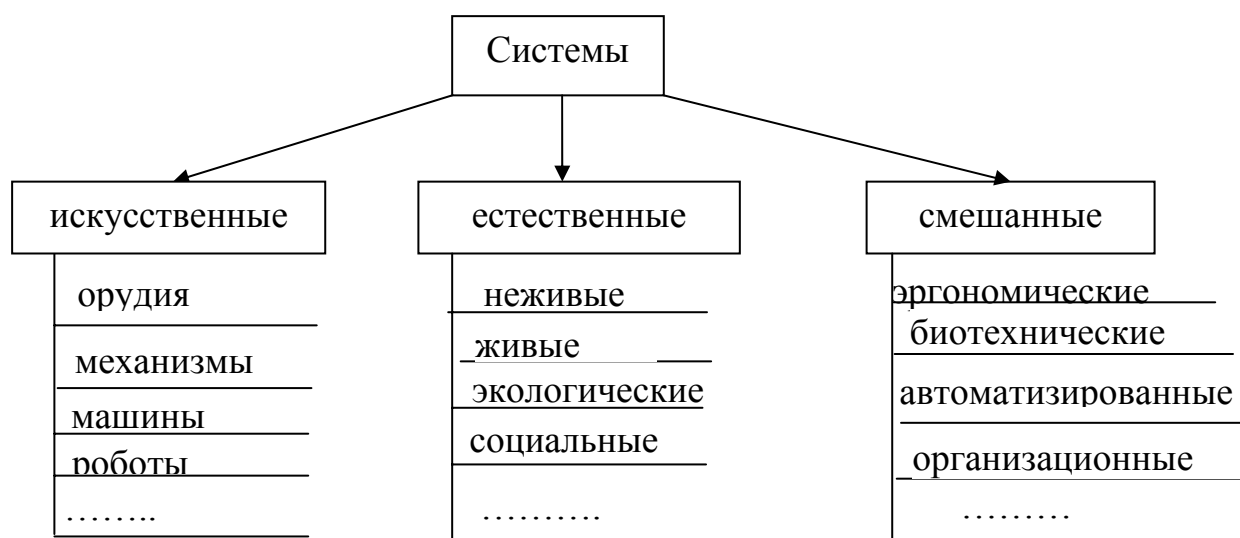


Рис.3.5. Классификация систем по происхождению

Поскольку классификация – это модель, то она, как и всякая модель, носит целевой характер: новые цели порождают и новые классификации. Чтобы как-то упорядочить эту классификацию, рассмотрим общую функциональную схему системы управления, приведенную на рис. 3.6.

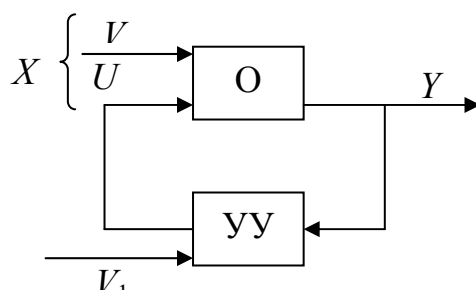


Рис.3.6. Функциональная схема систем управления

На этой схеме выделен объект управления O и управляющее устройство $УУ$, вырабатывающее управляющие воздействия U . И методы нахождения управления U , и способы его осуществления, и сам результат управления зависит от того, что известно об объекте и как эти знания используются управляющим устройством. Рассматривая разные аспекты вышеизложенного, можно строить разные классификации систем. Например, по типам входных, выходных переменных и переменных состояния. На рис. 3.7 приведен фрагмент такой классификации:

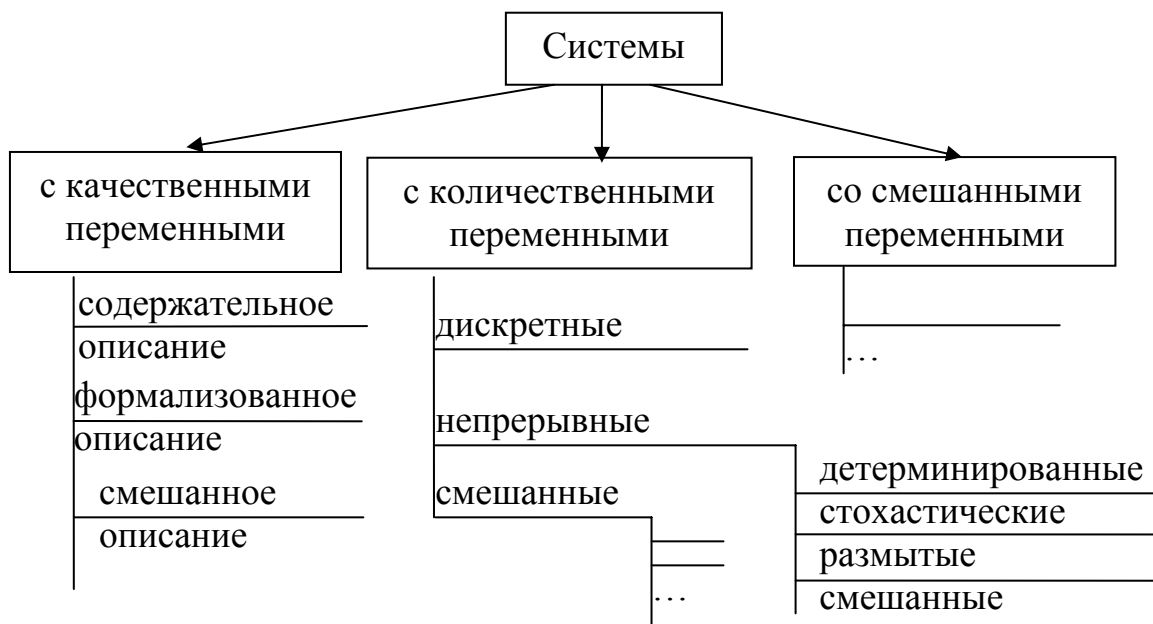


Рис.3.7. Классификация систем по описанию переменных

Принципиально разных подходов требуют переменные, описываемые количественно и качественно, что и приводит к первому уровню классификации. Могут быть ситуации, когда часть переменных описывается количественными, а часть – качественными характеристиками, поэтому и появляется класс смешанного описания. Для систем с качественными переменными следующий уровень классификации подразумевает описание переменных на уровне естественного языка и на более формализованном уровне (например, на уровне предикатных переменных). Возможно также и смешанное описание. Для систем с количественным описанием переменных разного подхода требуют переменные непрерывные и дискретные. Выделен также подкласс дискретно-непрерывного описания переменных. Для третьего класса систем (со смешанным описанием переменных) второй уровень классификации является объединением подклассов систем двух первых ветвей и на рис. 3.7 не показан. Третий уровень классифика-

ции можно принять одинаковым для всех подклассов второго уровня. На рис. 3.7 он показан только для одного подкласса.

3.5.2. Типы операторов систем

Можно приводить классификацию систем по типам связей между входными и выходными переменными, то есть по типу оператора системы (отображения выхода λ). Различие информации об этом операторе дает основание для первого уровня классификации (см. рис. 3.8). Отсутствие информации приводит к классу на уровне “черного ящика”. Дальше в этом классе деления нет. Самые общие сведения об отображении λ , когда это отображение нельзя привести к параметризованному виду, позволяет выделить непараметризованный класс. Дальнейшая классификация непараметризованных операторов связана с типом имеющейся информации.

Если преобразование известно с точностью до параметров – получаем следующий класс систем. На рис. 3.8 в качестве примера продолжена классификация одной из ветвей этого класса.

Наконец, если оператор известен точно, то получаем четвертый класс систем – “белый ящик” с дальнейшим делением этого класса аналогичным образом, как и для параметрических операторов.

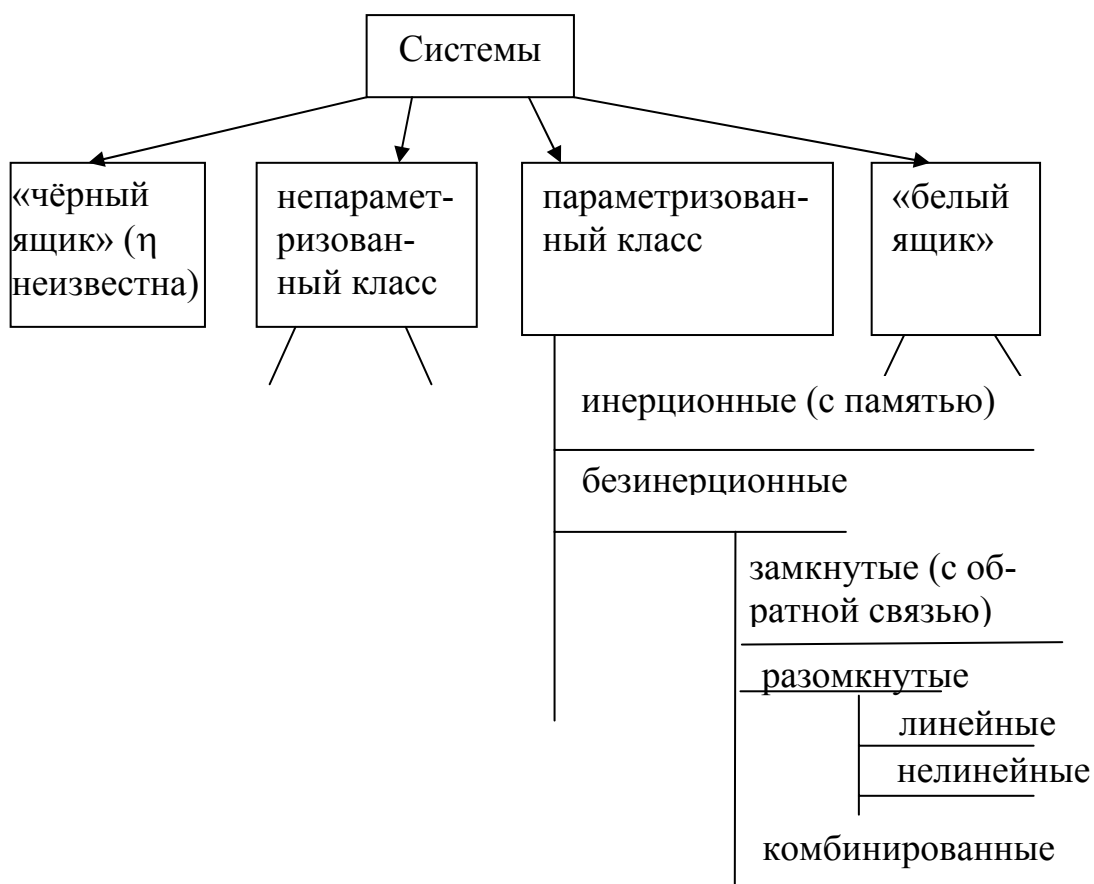


Рис.3.8. Классификация систем по типам операторов

3.5.3. Типы способов управления

Классификация по этому принципу приведена на рис. 3.9.

В зависимости от того, входит управляющее устройство в систему или является внешним по отношению к ней, выделен первый уровень классификации.

Для пояснения второго уровня классификации представим движение динамической системы в виде траектории, например, в пространстве состояний Q^n . Независимо от того, включен ли в систему управляющий блок или нет, выделяются четыре основных способа управления. Первый (простейший) случай соответствует полной известности заданной, требуемой траектории при точном программном управлении $U_0(t)$. В этом случае управление осуществляется без контроля за состоянием объекта.

Если движение системы отличается от заданного (что может быть при отличии неуправляемых переменных $V_0(t)$ от ранее предполагаемых, или действуют неучитываемые факторы), то возникает необходимость контроля за состоянием объекта. В этом случае мы приходим ко второму подклассу второго уровня.

Для таких систем, в которых точную требуемую траекторию движения задать невозможно, а известна только конечная целевая окрестность, предусмотрен третий подкласс второго уровня. Для систем этого подкласса осуществляется подстройка параметров заранее непредсказуемым образом.

Наконец, когда никакой подстройкой параметров невозможно достичь целевой области, приходится идти на изменение структуры системы (то есть, по сути, заменять систему другой системой). В этом случае (это четвертый подкласс) имеет место перебор разных систем (имеющих одинаковые выходы Y), создаваемых, тем не менее, не произвольно, а с учетом наличных средств.

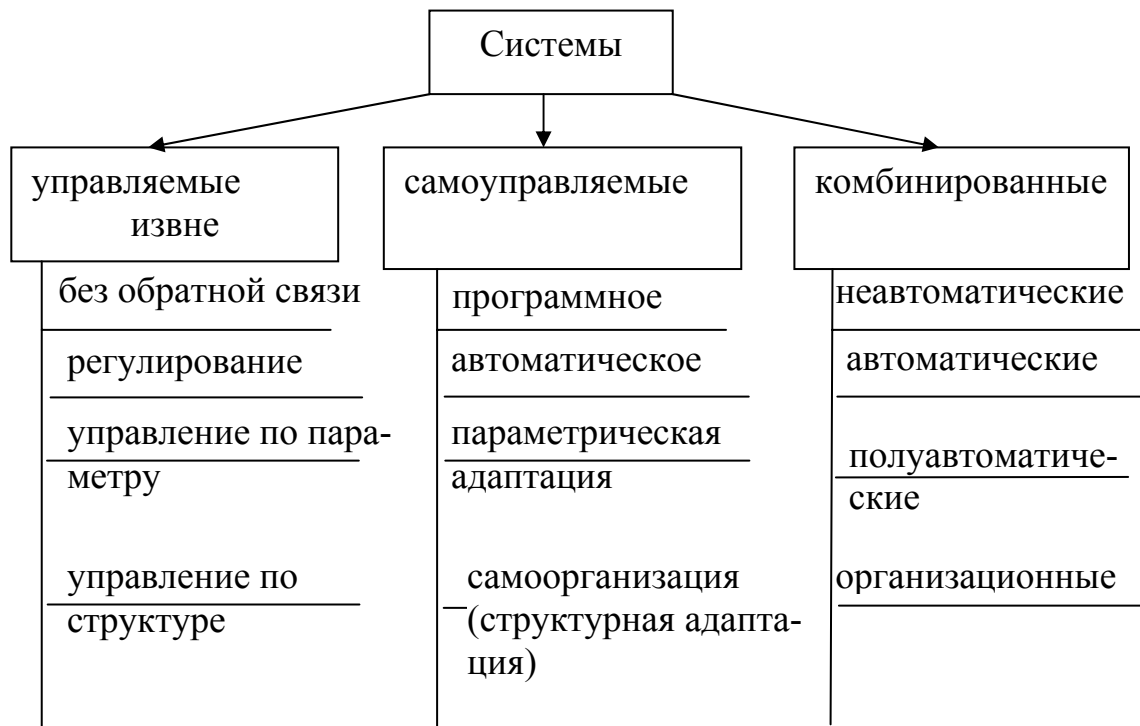


Рис.3.9. Классификация систем по способу управления

3.5.4. Ресурсное обеспечение

Ясно, что для обеспечения определенного управления в любой системе требуются ресурсы: материальные, энергетические, информационные. Причем в зависимости от достаточности или недостатка этих видов ресурсов возможны принципиально различные случаи систем. Это и дает основания для классификации, приведенной на рис. 3.10.

Энергетические затраты на выработку управления, как правило, малы по сравнению с энергопотреблением объекта управления, и в этом случае их просто не принимают во внимание в классе обычных систем. Но в некоторых случаях управляемая и управляющая части могут потреблять соизмеримое количество энергии и питаться от одного источника. При этом возникает нетривиальная задача перераспределения ограниченной энергии между этими частями, и мы приходим к классу энергокритичных систем.

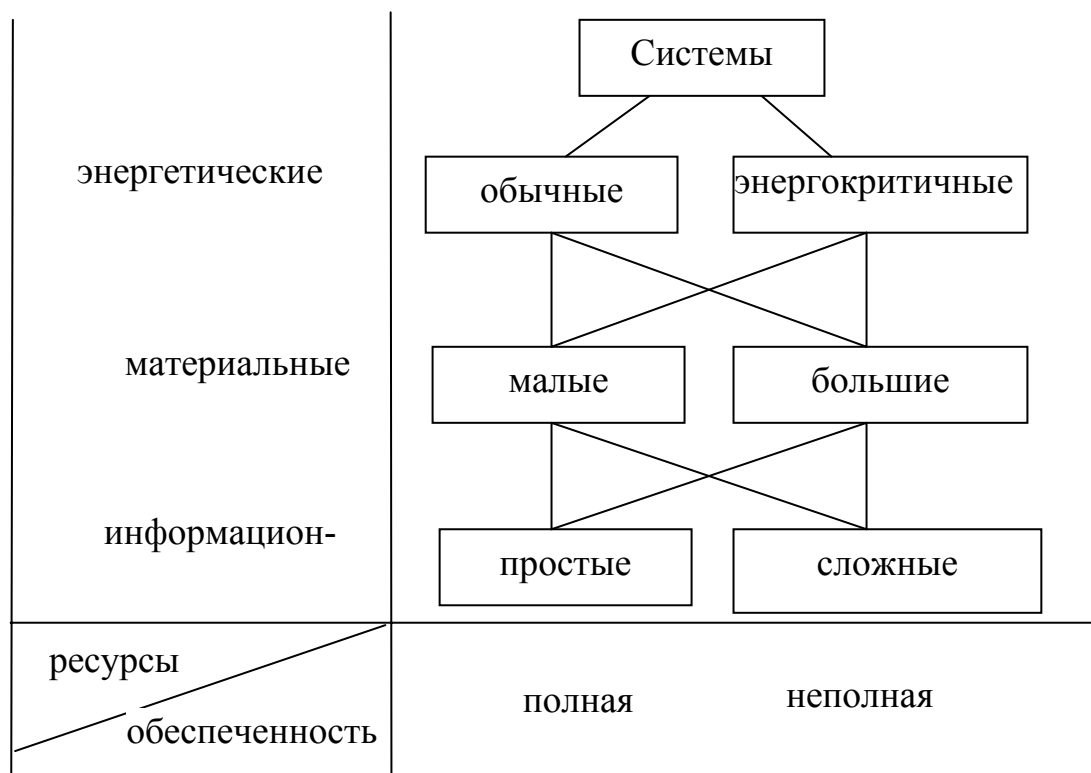


Рис.3.10. Классификация систем по ресурсообеспеченности

Материальные затраты на функционирование систем также могут создавать определенные ограничения. Например, в случае управления объектом большой размерности с помощью ЦВМ такими ограничениями могут быть объем оперативной памяти и быстродействие. В соответствии с этим большими можно назвать системы, моделирование которых затруднено вследствие их большой размерности. Перевод систем из больших в малые можно осуществить двумя способами. Это либо применение более мощной вычислительной базы, либо декомпозиция многомерной задачи на задачи меньшей размерности (если это возможно).

Третий тип ресурсов – информационный. Если информации о системе достаточно (а признаком этого служит успешность управления), то систему можно назвать простой. Если же управление, выработанное на основе имеющейся информации об объекте, приводит к неожиданным, непредвиденным или нежелательным результатам, то систему можно интерпретировать как сложную. Поэтому сложной системой можно назвать систему, в модели которой недостаточно информации для эффективного управления. Два способа можно предложить для перевода системы из подкласса сложных в подкласс простых. Первый состоит в выяснении конкретной причины сложности, получении недостающей информации и включении ее в модель системы. Второй способ связан со сменой цели, что в технических системах неэффективно, но в отношениях между людьми часто является единственным выходом.

Поскольку перечисленные виды ресурсов являются более или менее независимыми, возможно самое различное сочетание между подклассами этой классификации.

Как и все предыдущие, приведенная классификация может быть при необходимости развита: либо при более подробном рассмотрении видов ресурсов, либо в результате введения большего числа градаций степени обеспеченности ими.

4. АВТОМАТНОЕ ОПИСАНИЕ СИСТЕМ. ТЕОРИЯ КОНЕЧНЫХ АВТОМАТОВ

В том случае, если множества входных X , выходных Y и внутренних переменных Q являются конечными, мы приходим к так называемому автоматному описанию системы. Для этого случая разработана достаточно разветвленная, хорошо формализованная *теория конечных автоматов*. Конечными автоматами можно описывать любые цифровые устройства и элементы. Анализ и синтез цифровых устройств также успешно может быть осуществлен с применением теории конечных автоматов. Для изучения этого раздела достаточно знать методы формальной логики, теорию множеств, теорию графов и основы абстрактной алгебры.

Естественно, что и пользоваться при этом мы будем понятиями, определениями и терминологией (то есть профессиональным языком), принятыми в соответствующих разделах дискретной математики.

4.1. Основные понятия. Способы задания автоматов

4.1.1. Определение абстрактного автомата

Пусть $X = \{x_1, x_2, \dots, x_m\}$ и $Y = \{y_1, y_2, \dots, y_k\}$ – два произвольных множества элементов, которые будем называть *алфавитами*, а их элементы – *буквами алфавита*. Конечную упорядоченную последовательность букв назовем *словом* в данном алфавите. Обозначим X^* и Y^* – множества всех слов в алфавитах X и Y соответственно. Тогда произвольное преобразование дискретной информации можно задать как однозначное отображение S множества слов X^* во множество слов Y^* . Отображение S называется алфавитным отображением или алфавитным оператором, а алфавиты X и Y – входным и выходным алфавитами оператора S . Каждому входному слову

$\mathbf{x} = x_{i_1} x_{i_2} \dots x_{i_k}$ оператор S сопоставляет выходное слово $y = y_{i_1} y_{i_2} \dots y_{i_k}$. Поэтому для каждого $\mathbf{x} \in X^*$ существует свое $\mathbf{y} \in Y^*$, такое, что $\mathbf{y} = S(\mathbf{x})$. То есть S есть функция, область определения которой X^* , а область значений – Y^* .

В общем случае отображение S может быть частичным, то есть не всюду определенным. Это позволяет рассматривать отображение S как оператор в одном и том же расширенном алфавите $Z = X \cup Y$. Частичное отображение S множества Z^* в себя, определенное на словах, состоящих только из $\{x_1 \dots x_m\}$, можно выбрать таким образом, что оно будет совпадать с отображением S множества X^* в Y^* . Любой абстрактный автомат реализует некоторый оператор S или *индуцирует* некоторое отображение S .

Условия, накладываемые на автоматные отображения, рассмотрим несколько позже, а сейчас отметим ряд допущений, связанных с понятием абстрактного автомата:

а) наличие произвольного числа отличных друг от друга состояний автомата и свойство мгновенного перехода из одного состояния в другое;

б) переход из одного состояния в другое оказывается возможным не ранее, чем через некоторый промежуток времени Δ ($\Delta > 0$ – интервал дискретности) после предыдущего перехода;

в) число различных входных и выходных сигналов конечно;

г) входные сигналы – причина перехода автомата из одного состояния в другое, а выходные сигналы – реакция автомата на входные сигналы и относятся они к моментам времени, определенным соответствующим переходом автомата из состояния в состояние.

Учитывая это, можно интерпретировать автомат как устройство, работающее в дискретном времени $t' = n \times \Delta$ ($n \in N_0$). На каждый входной сигнал $x(t)$ автомат реагирует выходным сигналом $y(t)$, где $t = \frac{t'}{\Delta}$ – норми-

рованное время. Связь между входом и выходом определяется текущим состоянием автомата q и задается функцией выхода $y=\lambda(q, x)$, а переход автомата из одного состояния в другое – функцией переходов $q=\delta(q, x)$.

Теперь можно дать определение абстрактного автомата. Это пятерка объектов:

$$A = (X, Y, Q, \lambda, \delta),$$

где $X = \{x_1, x_2, \dots, x_m\}$ - входной алфавит или множество входных состояний;

$Y = \{y_1, y_2, \dots, y_k\}$ - выходной алфавит или множество выходных состояний;

$Q = \{q_1, q_2, \dots, q_n\}$ - множество внутренних состояний; $\delta: Q \times X \rightarrow Q$ – функция перехода; $\lambda: Q \times X \rightarrow Y$ – функция выхода.

Если $|Q| < \infty$, то мы имеем дело с *конечным* автоматом, если Q – множество счетное – то с бесконечным. Если начальное состояние зафиксировано (например q_1), то автомат называется инициальным. Таким образом, по неинициальному автомату можно n способами задать инициальный автомат.

В приведенном определении ничего не сказано о времени, ни о непрерывном, ни о дискретном. Таким образом, представление абстрактного автомата или его интерпретация как некоторого устройства, на которые в дискретные моменты времени поступают сигналы и в эти же моменты времени выдаются выходные сигналы, позволяет только более наглядно представить его работу.

С другой стороны, описание реального устройства, функционирующего в реальном времени, в виде автомата является абстрактной моделью. Следовательно, как всякая модель, автоматная модель является упрощенной (фактически время перехода не нулевое), конечной (описывает систему только на уровне входов, выходов и состояний) и приближенной

(реальное поведение системы может отличаться от модельного за счет помех, непредусмотренных внешних воздействий и т.п.).

Покажем теперь, каким образом определить отображение S , индуцируемое заданным конечным автоматом. Возьмем интерпретацию автомата как устройства, функционирующего в дискретном времени. Предположим, что автомат инициальный и задано начальное состояние q_1 . В каждый момент времени, отличный от нулевого, на вход автомата поступает входной сигнал $x(t)$ – произвольная буква входного алфавита $x(t) \in X$, а на выходе возникает некоторый выходной сигнал $y(t)$ – буква выходного алфавита $y(t) \in Y$. Для данного автомата его функции δ и λ могут быть определены не только на множестве X всех входных букв, но и на множестве X^* всех входных слов. Действительно, для любого входного слова $\mathbf{x} = x_{i_1} x_{i_2} \dots x_{i_k}$

$$\delta(q_1, \mathbf{x}) = \delta(q_1, x_{i_1} x_{i_2} \dots x_{i_k}) = \delta\left(\delta\left(\dots \delta\left(q_1, x_{i_1}\right), x_{i_2}\right), x_{i_k}\right), \quad (4.1.1)$$

или, используя индуктивное определение:

для каждой входной буквы x_j функция $\delta(q_i, x_j)$ первоначально задана,

для любого слова $\mathbf{x} \in X^*$ и любой буквы $x_j \in X$

$$\delta(q_i, \mathbf{x}x_j) = \delta(\delta(q_i, \mathbf{x}), x_j). \quad (4.1.2)$$

С помощью такой расширенной функции δ можно также индуктивно определить и λ :

$$\lambda(q_i, \mathbf{x}x_j) = \lambda(\delta(q_i, \mathbf{x}), x_j). \quad (4.1.3)$$

Появление на входе конечной последовательности $x(1) = x_{i_1}$, $x(2) = x_{i_2}, \dots, x(l) = x_{i_l}$, то есть входного слова $\mathbf{X} = x_{i_1} x_{i_2} \dots x_{i_l}$ на основании известных функций λ и δ , вызовет появление на выходе однозначной по-

следовательности $y(1) = y_{j_1}, y(2) = y_{j_2}, \dots, y(l) = y_{j_l}$, которая соответствует выходному слову

$$y = y_{j_1} y_{j_2} \dots y_{j_l} = \lambda(q_1, x_{i_1}) \lambda(q_1, x_{i_1} x_{i_2}) \dots \lambda(q_1, x_{i_1} \dots x_{i_l}).$$

Соотнося каждому входному слову соответствующее ему выходное, получим искомое отображение, которое и является автоматным отображением, или автоматным оператором. Если результатом применения оператора к слову x будет выходное слово y , то это обозначается так: $S(q_1, x) = y$ или $S(x) = y$.

Автоматное отображение можно определить и индуктивно:

$$S(q_i, x_j) = \lambda(q_i, x_j); \quad (4.1.4)$$

$$S(q_i, x x_j) = S(q_i, x) \lambda(\delta(q_i, x), x_j). \quad (4.1.5)$$

Автоматное отображение обладает двумя свойствами, непосредственно следующими из определения:

слова x и $y = S(x)$ имеют одинаковую длину;

если $x = x_1 x_2$ и $S(x_1 x_2) = y_1 y_2$, где $|x_1| = |y_1| = i$, то $S(x_1) = y_1$. Иначе говоря, образ отрезка длины i равен отрезку образа той же длины.

Свойство второе означает, что автоматные операторы – операторы без предвосхищения, то есть операторы, которые перерабатывают слово слева направо, не «заглядывая» вперед.

Эти два свойства были бы достаточными для автоматного отображения, если бы речь шла о бесконечных автоматах, то есть автоматах с бесконечным числом состояний. Для конечных автоматов этих условий, как мы дальше увидим, недостаточно.

Таким образом, одна из интерпретаций абстрактного автомата – это некоторое цифровое вычислительное, управляющее или преобразовательное устройство. Входная буква – это входной сигнал (комбинация сигналов на всех входах устройства), входное слово – последовательность входных сигналов, поступающих в дискретные моменты времени $t = 1, 2, 3, \dots$

Выходное слово – последовательность выходных сигналов, выдаваемых автоматом, внутреннее состояние автомата – комбинация состояний запоминающих элементов устройства.

Такая интерпретация, безусловно, верна, но она не требуется ни для определения автомата, ни для построения соответствующей теории конечных автоматов. Все, что действительно важно в абстрактной теории автоматов – это работа со словами при конечной памяти.

4.1.2. Задание автоматов

Поскольку функции δ и λ определены на конечных множествах, то их можно задать таблицами. Обычно две таблицы (для функции δ и для функции λ) сводят в одну таблицу $\delta \times \lambda : Q \times X \rightarrow Q \times Y$ и называют такую таблицу *автоматной таблицей*, или таблицей переходов автомата. В этой таблице на пересечении строки с входной буквой x_i и столбца с состоянием q_j стоит пара – состояние q_l , в которое переходит автомат из состояния q_j по входной букве x_i , и выходная буква y_k , которая при этом выдается автоматом.

Часто вместо автоматной таблицы для задания автомата используют так называемую *матрицу соединений автомата*. Это матрица $n \times n$, строки и столбцы которой соответствуют различным состояниям автомата. На пересечении q_i -й строки и q_j -го столбца стоит буква (или дизъюнкция букв) входного алфавита $x_l \in X$, вызывающая переход автомата из состояния q_i в состояние q_j , и в скобках (можно через запятую, тире или слеш) – буква (или дизъюнкция букв) выходного алфавита $y_k \in Y$, которая появляется при этом на выходе автомата. Если ни одна из букв входного алфавита не переводит состояние q_i в q_j , то ставится прочерк или ноль. В любой строке

каждая буква входного алфавита встречается не более одного раза (условие однозначности переходов).

Еще один способ задания автоматов – ориентированный мультиграф, называемый *графом переходов*, или диаграммой переходов. Вершины графа переходов соответствуют состояниям автомата. Если $\delta(q_i, x_j) = q_k$ и $\lambda(q_i, x_j) = y_l$, то из вершины q_i в вершину q_k ведет ребро, на котором написаны пара $x_j y_l$. Для любого графа переходов выполняются следующие условия корректности:

а) для любой входной буквы x_j имеется ребро, выходящее из q_i , на котором написано x_j (условие полноты);

б) любая буква x_j встречается только на одном ребре, выходящем из вершины q_i (условие непротиворечивости, или детерминированности).

На графе переходов наглядно представимы все функции, определяемые формулами (4.1.1) - (4.1.4). Если зафиксирована вершина q_i , то всякое слово $\mathbf{x} = x_{i_1} x_{i_2} \dots x_{i_k}$ однозначно определяет путь длины k из этой вершины (обозначим его q_i, \mathbf{x}), на k ребрах которого написаны $x_{i_1} x_{i_2} \dots x_{i_k}$. Поэтому $\delta(q_i, \mathbf{x})$ – это последняя вершина пути q_i, \mathbf{x} ; $\lambda(q_i, \mathbf{x})$ – выходная буква, написанная на последнем ребре пути q_i, \mathbf{x} , а отображение $S(q_i, \mathbf{x})$ – слово, образованное выходными буквами на k ребрах пути q_i, \mathbf{x} .

Пример 4.1. Пусть автомат задан своей автоматной таблицей:

Таблица 4.1

$q \backslash x$	x_1	x_2
1	2, y_1	3, y_3
2	2, y_2	3, y_1
3	1, y_1	2, y_2

Здесь, и в дальнейшем, если это не вызовет разночтений, внутренние состояния обозначены своими индексами, т.е. алфавит состояний – это $Q = \{1, 2, 3\}$. Входной алфавит автомата $X = \{x_1, x_2\}$, выходной алфавит $Y = \{y_1, y_2, y_3\}$.

Матрица соединений данного автомата приведена ниже:

Таблица 4.2

$q \backslash q$	1	2	3
1	0	x_1, y_1	x_2, y_3
2	0	x_1, y_2	x_2, y_1
3	x_1, y_1	x_2, y_2	0

На рис. 4.1 представлен граф (состояний) автомата.

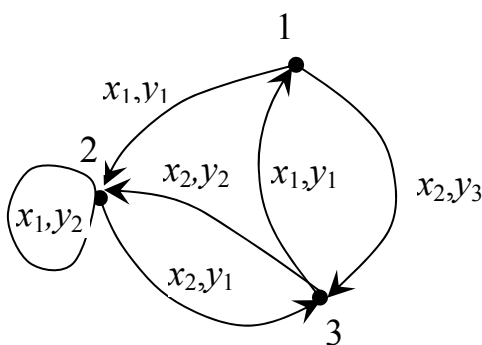


Рис 4.1. Переходной граф состояний

Если на вход автомата, находящегося в состоянии 1, поступит, например, слово $x = x_1 x_2 x_2 x_1 x_2 x_2$, то на выходе появится слово $y = y_1 y_1 y_2 y_2 y_1 y_2$, а автомат перейдет в состояние 2.

Пример 4.2. Граф, представленный на рис. 4.2, нельзя интерпретировать как некоторый автомат, поскольку в этом графе нарушено условие автоматности: из вершины 2 выходят два ребра с одной буквой a .

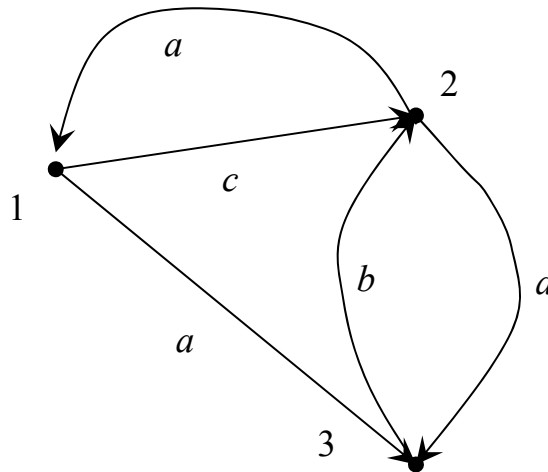


Рис. 4.2. Граф, не являющийся автоматом

4.2. Виды автоматов и их свойства

Дадим несколько определений, которые понадобятся для дальнейшего изложения.

Состояние q_j называется *достижимым* из состояния q_i , если существует входное слово x , такое, что $\delta(q_i, x) = q_j$. Автомат называется *сильно связанным*, если из любого его состояния достижимо любое другое состояние.

4.2.1. Автономные автоматы

Автомат называется *автономным по входу*, если его входной алфавит состоит из одной буквы: $X = \{x\}$. Все входные слова у такого автомата имеют вид $xx \dots x$.

Теорема 4.2.1. Любое достаточно длинное выходное слово автономного по входу автомата с n состояниями является периодическим (возможно с предпериодом), причем длины периода и предпериода не превосходят n . Оно имеет вид $\sigma\omega\omega\ldots\omega\omega_1$, где $0 \leq |\sigma| \leq n, 1 \leq |\omega| \leq n$, ω_1 - начальный отрезок ω .

Доказательство. Так как в графе автономного по входу автомата из каждой вершины выходит одно ребро, то его сильно связанные подграфы могут быть только простыми циклами, из которых нет выходных ребер. Поэтому в компоненте связности может быть только один цикл. Остальные подграфы компоненты связности – это деревья, подвешенные к циклу и ориентированные в его сторону. Что и требовалось доказать.

Пример 4.3. Автомат задан автоматной таблицей:

Таблица 4.3

q	x
1	3,0
2	4,0
3	4,0
4	7,0
5	4,2
6	5,0
7	6,1

Граф автомата приведен на рис. 4.3. Входная буква на ребрах графа не показана.

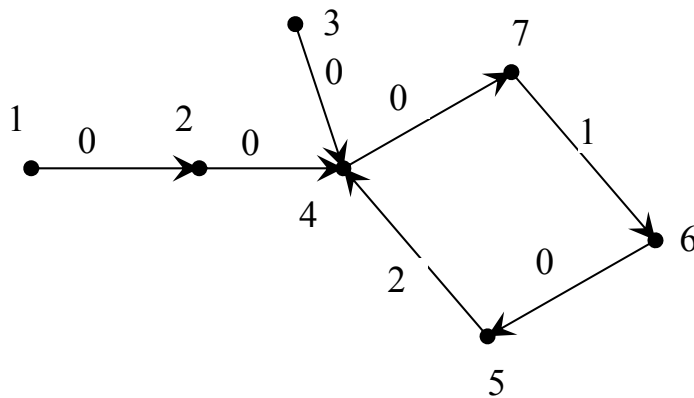


Рис.4.3. Граф автономного автомата

Пусть автомат первоначально находится в состоянии 2, и на вход поступает слово xxxxxxxxxxxx. На выходе будет слово 00102010201, где предпериод $\sigma=0$, период $\omega=0102$, начальный отрезок $\omega_1=01$.

Из произвольного автомата с входным алфавитом $X=\{x_1, \dots, x_m\}$ может быть построено m различных автономных по входу автоматов исключением из графа переходов автомата всех ребер, кроме ребер с выбранной буквой x_i ($i=1, \dots, m$).

Аналогично, автомат называется автономным по выходу, если его выходной алфавит состоит из одной буквы $Y=\{y\}$. Автономный по выходу автомат получается из произвольного автомата с выходным алфавитом $Y=\{y_1, y_2, \dots, y_k\}$ исключением из графа переходов ребер со всеми выходными буквами кроме выбранной буквы y_i .

Пример 4.4. Возьмем автомат из примера 4.1. Его автоматная таблица приведена ниже:

$q \backslash x$	x_1	x_2
1	2, y_1	3, y_3
2	2, y_2	3, y_1
3	1, y_1	2, y_2

Таблица автономного автомата по входной букве, например x_2 , получится удалением всех столбцов, кроме столбца с буквой x_2 :

$q \backslash x$	x_2
1	3, y_3
2	3, y_1
3	2, y_2

Таблица автономного автомата по выходной букве, например y_1 , получится удалением всех элементов исходной таблицы, кроме элементов с буквой y_1 :

$q \backslash x$	x_1	x_2
1	2, y_1	-
2	-	3, y_1
3	1, y_1	-

4.2.2. Автоматы синхронные и асинхронные

В синхронных автоматах переход из одного состояния в другое осуществляется через равные промежутки времени, задаваемые в реальных устройствах генератором тактовых импульсов. Другими словами, синхронный автомат реагирует на каждую букву входного алфавита.

В асинхронном автомате его внутреннее состояние может меняться только при изменении входного состояния. В результате этого изменения автомат всегда приходит в конечном итоге в некоторое устойчивое полное состояние, то есть в такое полное состояние, в котором автомат остается до тех пор, пока не изменится его входное состояние (полное состояние – это совокупность входного и внутреннего состояний).

Полагают также, что новое изменение входа не может произойти до того, как автомат перейдет в устойчивое полное состояние.

В итоге моменты перехода асинхронного автомата из состояния в состояние зависят от значения входа. Понятно, что при этом теряет смысл рассмотрение входных слов, содержащих одинаковые соседние буквы.

Сформулированные для асинхронного автомата условия налагают некоторые ограничения на таблицу переходов $\|\delta_{ij}\|$.

Чтобы было понятнее, рассмотрим вначале усиленный вариант этих условий, когда любое полное состояние автомата связано с некоторым устойчивым состоянием прямым, непосредственным переходом. Это означает, что если некоторый элемент δ_{ij} автоматной таблицы имеет значение q_k , то это же значение должен иметь и элемент δ_{kj} .

Такому требованию удовлетворяет, например, следующая таблица переходов:

Таблица 4.4

$q \backslash x$	x_1	x_2	x_3	x_4
1	1	1	1	5
2	1	2	2	2
3	3	2	4	5
4	3	2	4	5
5	3	5	4	5

Жирным шрифтом выделены элементы, соответствующие устойчивым полным состояниям.

В общем виде эти условия формулируются так: для любого элемента δ_{ij} таблицы переходов должна выполняться цепочка равенств

$$\delta_{ij} = k_1, \delta_{k_1j} = k_2, \dots, \delta_{k_{p-1}j} = k_p.$$

Это означает, что любое полное состояние (q_i, x_j) асинхронного автомата должно быть связано цепочкой переходов с некоторым устойчивым полным состоянием (q_{k_p}, x_j) .

На таблицу выходов $\|\lambda_{ij}\|$ асинхронного автомата каких-либо ограничений не налагают.

4.2.3. Автоматы Мили и автоматы Мура

Общее определение автомата, данное в разделе 4.1, задает так называемый *автомат Мили*. Характерной особенностью автомата Мили является то, что значение его выхода зависит от полного состояния, то есть как от внутреннего, так и от входного состояний. Другими словами, функция выхода λ является двуместной функцией $y(t) = \lambda(q(t-1), x(t))$.

В случае если функция выхода зависит только от внутреннего состояния, но не от входа, получаем автомат, носящий название *автомата Мура*. Для автомата Мура для любых q , x_i и x_j выполняется условие $\lambda(q, x_i) = \lambda(q, x_j)$, то есть функция выхода – одноместная. Часто ее в этом случае обозначают буквой μ и называют *функцией отметок*, так как она каждое состояние помечает вполне однозначно буквами выходного алфавита.

Для автомата Мура таблица выходов вырождается в один столбец, а автоматная таблица записывается с лишним столбцом. Матрица соединений также содержит лишний столбец.

Возможности этих двух видов автоматов совпадают, то есть для любого автомата Мили существует эквивалентный ему автомат Мура (и наоборот). Это утверждение можно сформулировать в виде теоремы:

Теорема 4.2.2. Для произвольного автомата Мили

$$S = (X, Q, Y, \delta, \lambda), \quad X = \{x_1, x_2, \dots, x_m\}, \quad Q = \{q_1, q_2, \dots, q_n\},$$

существует эквивалентный ему автомат Мура

$$S_M = (X_M, Q_M, Y_M, \delta_M, \mu).$$

Он может быть построен следующим образом: входной и выходной алфавиты исходного автомата Мили и эквивалентного автомата Мура совпадают $X_M = X$, $Y_M = Y$. Алфавит состояний Q_M содержит $m \cdot n + n$ состояний: $m \cdot n$ состояний q_{ij} ($i=1, \dots, n, j=1, \dots, m$), соответствующих парам (q_i, x_j) автомата S и n состояний q_{i0} ($i=1, \dots, n$). Функция δ_M определяется так: $\delta_M(q_{i0}, x_k) = q_{ik}$ ($i=1, \dots, n$), $\delta_M(q_{ij}, x_k) = q_{lk}$, где индекс l определяется функцией перехода автомата S : $\delta(q_i, x_j) = q_l$. Функция отметок $\mu(q_{i0})$ - не определена, а для остальных состояний $\mu(q_{ij}) = \lambda(q_i, x_j)$. Состояние q_{i0} ($i=1, \dots, n$) автомата S_M отождествляется с начальным состоянием q_i автомата S (если задан инициальный автомат).

Доказательство теоремы заключается в том, чтобы показать равенство автоматных отображений $S(q_i, \mathbf{x}) = S_M(q_{i0}, \mathbf{x})$ для любого состояния q_i и любого слова \mathbf{x} . Это делается индукцией по длине \mathbf{x} и предлагается проделывать самостоятельно.

Пример 4.5. Автомат Мили задан автоматной таблицей:

Таблица 4.5

$q \backslash x$	x_1	x_2
1	$2, y_1$	$3, y_1$
2	$2, y_2$	$3, y_2$
3	$1, y_3$	$2, y_1$

Для данного автомата число состояний $n=3$, число входных букв $m=2$. Построим эквивалентный автомат Мура. В соответствии с теоремой

4.2.2 число состояний эквивалентного автомата Мура составит $n \cdot m + n = 9$.

Полагая в формуле $\delta_M(q_{i0}, x_k) = q_{ik}$ $i=1,2,3$; $k=1,2$, получим

$$\begin{aligned}\delta_M(q_{10}, x_1) &= q_{11}, & \delta_M(q_{20}, x_1) &= q_{21}, & \delta_M(q_{30}, x_1) &= q_{31}, \\ \delta_M(q_{10}, x_2) &= q_{12}, & \delta_M(q_{20}, x_2) &= q_{22}, & \delta_M(q_{30}, x_2) &= q_{32}.\end{aligned}$$

Воспользовавшись формулой $\delta_M(q_{ij}, x_k) = q_{lk}$ и учитывая, что индекс l определяется из соотношения $\delta(q_i, x_j) = q_l$ табл. 4.5, имеем

$$\begin{aligned}\delta_M(q_{11}, x_1) &= q_{21}, & \delta_M(q_{21}, x_1) &= q_{21}, & \delta_M(q_{31}, x_1) &= q_{11}, \\ \delta_M(q_{11}, x_2) &= q_{22}, & \delta_M(q_{21}, x_2) &= q_{22}, & \delta_M(q_{31}, x_2) &= q_{12}, \\ \delta_M(q_{12}, x_1) &= q_{31}, & \delta_M(q_{22}, x_1) &= q_{31}, & \delta_M(q_{32}, x_1) &= q_{21}, \\ \delta_M(q_{12}, x_2) &= q_{32}, & \delta_M(q_{22}, x_2) &= q_{32}, & \delta_M(q_{32}, x_2) &= q_{22}.\end{aligned}$$

Далее находим функцию отметок по формуле $\mu(q_{ij}) = \lambda(q_i, x_j)$ и составляем автоматную таблицу автомата Мура. Последняя будет выглядеть так:

Таблица 4.6

	x_1	x_2	μ
q_{10}	q_{11}	q_{12}	-
q_{20}	q_{21}	q_{22}	-
q_{30}	q_{31}	q_{32}	-
q_{11}	q_{21}	q_{22}	y_1
q_{12}	q_{31}	q_{32}	y_1
q_{21}	q_{21}	q_{22}	y_2
q_{22}	q_{31}	q_{32}	y_2
q_{31}	q_{11}	q_{12}	y_3
q_{32}	q_{21}	q_{22}	y_1

Обратное (получение автомата Мили по автомату Мура) очевидно и не вызывает трудностей.

Пример 4.6. Пусть дан автомат Мура $A = (X, Q, Y, q_1 \in Q, \delta, \mu)$, где $X = \{x_1, x_2\}$, $Q = \{1, 2, 3\}$, $Y = \{y_1, y_2, y_3\}$, причем

$$\begin{aligned}\delta(1, x_1) &= 2, & \delta(1, x_2) &= 3, \\ \delta(2, x_1) &= 3, & \delta(2, x_2) &= 2, \\ \delta(3, x_1) &= 1, & \delta(3, x_2) &= 2, \\ \mu(1) &= y_2, & \mu(2) &= y_1, & \mu(3) &= y_2.\end{aligned}$$

По исходным данным легко воспроизвести автоматную таблицу (таблицу переходов):

Таблица 4.7

	x_1	x_2	μ
1	2	3	y_2
2	3	2	y_1
3	1	2	y_2

Построим эквивалентный автомат Мили B . Используя табл. 4.7, строим обычную функцию выхода $\lambda(q, x)$, определяющую автомат Мили $B = (X, Q, Y, q_1 \in Q, \delta, \lambda)$:

$$\begin{aligned}\lambda(1, x_1) &= y_1, & \lambda(1, x_2) &= y_2, \\ \lambda(2, x_1) &= y_2, & \lambda(2, x_2) &= y_1, \\ \lambda(3, x_1) &= y_2, & \lambda(3, x_2) &= y_1.\end{aligned}$$

Автоматная таблица построенного автомата Мили выглядит следующим образом:

Таблица 4.8

	x_1	x_2
1	$2, y_1$	$3, y_2$
2	$3, y_2$	$2, y_1$
3	$1, y_2$	$2, y_1$

Можно проверить, что автомат Мили B , заданный табл. 4.8, индуцирует такое же автоматное отображение, что и автомат Мура A , определяемый табл. 4.7.

Таким образом, при исследовании автоматов достаточно пользоваться только автоматом Мура. Это в некоторых случаях удобнее потому, что автомат Мура можно рассматривать как автомат без выходов, состояния которого различным образом отмечены. Можно считать, что таких отметок вообще две (например 0 и 1) и они делят состояния на два класса, один из которых можно назвать заключительным. Это позволяет дать еще одно определение абстрактного автомата – *автомата без выходов* $S = (X, Q, Y, \lambda, q_1, F)$, где $F \subseteq Q$ – множество заключительных состояний, а q_1 – начальное состояние автомата.

Вспоминая понятие автономного автомата, можно сказать, что автомат Мили может быть представлен как совокупность автономных автоматов по входным и выходным буквам. В случае автоматов Мура имеет смысл говорить об автономных автоматах только по входным буквам.

4.2.4. Автоматы первого и второго рода

Вспомним интерпретацию автомата как некоторого устройства, работающего в дискретном времени. Первопричиной появления выходного сигнала и изменения состояния является входной сигнал. Следовательно, выходной сигнал $y(t)$ всегда появляется после входного сигнала $x(t)$. Однако относительно времени t перехода автомата из состояния $q(t-1)$ в состояние $q(t)$ выходной сигнал может появиться либо раньше, либо позже этого момента времени. В первом случае уравнения, описывающие работу автомата, будут следующие:

$$\begin{aligned} q(t) &= \delta(q(t-1), x(t)), \\ y(t) &= \lambda(q(t-1), x(t)), \end{aligned} \quad (4.2.1)$$

а автомат будет именоваться *автоматом первого рода*.

Во втором случае получаем *автомат второго рода* с уравнениями:

$$\begin{aligned} q(t) &= \delta(q(t-1), x(t)), \\ y(t) &= \lambda(q(t), x(t)). \end{aligned} \quad (4.2.2)$$

В уравнениях (4.2.1) и (4.2.2) функция λ называется либо обычной (для автоматов первого рода), либо сдвинутой (для автоматов второго рода) функцией выхода.

Установим взаимосвязь между автоматами первого и второго рода. Пусть дан автомат второго рода $S = (X, Q, Y, \delta, \lambda)$. Запишем функцию переходов $\delta(q, x)$ и сдвинутую функцию выхода $\lambda(q, x)$:

$$\begin{aligned} y(t) &= \lambda(q(t), x(t)), \\ q(t) &= \delta(q(t-1), x(t)). \end{aligned}$$

Подставим в первое уравнение $q(t)$, определяемое вторым уравнением. Тогда получим уравнение

$$y(t) = \lambda(\delta(q(t-1), x(t)), x(t)) = \lambda'(q(t-1), x(t)),$$

определяющее обычную функцию выхода $\lambda'(q, x)$, которая характеризует автомат первого рода. Таким образом, подставляя в сдвинутую функцию выхода $\lambda(q, x)$ автомата второго рода функцию переходов $\delta(q, x)$, получаем автомат первого рода $S' = \{X, Q, Y, \delta, \lambda'\}$, который индуцирует то же самое автоматное отображение, что и автомат S . Такое сведение автомата второго рода к эквивалентному автомату первого рода называется интерпретацией автомата второго рода автоматом первого рода.

Пример 4.7. Пусть задан автомат $A = (X, Q, Y, q_1 \in Q, \delta, \lambda)$, где $X = \{x_1, x_2, x_3\}$, $Q = \{1, 2, 3, 4\}$, $Y = \{y_1, y_2\}$, а автоматная таблица следующая:

Таблица 4.9

$q \backslash x$	x_1	x_2	x_3
1	2, y_1	4, y_1	1, y_2
2	1, y_2	3, y_1	4, y_1
3	1, y_1	4, y_2	2, y_2
4	4, y_2	1, y_1	3, y_1

Предположим, что автомат A является автоматом первого рода. Тогда функция выхода $\lambda(q, x)$, полученная из табл. 4.9 и представленная в табл. 4.10, будет являться обычной функцией выхода.

Таблица 4.10

$q \backslash x$	x_1	x_2	x_3
1	y_1	y_1	y_2
2	y_2	y_1	y_1
3	y_1	y_2	y_2
4	y_2	y_1	y_1

Функция переходов $\delta(q, x)$, выделенная из табл. 4.9, приведена в табл. 4.11.

Таблица 4.11

$q \backslash x$	x_1	x_2	x_3
1	2	4	1
2	1	3	4
3	1	4	2
4	4	1	3

Если на вход автомата, находящегося первоначально в состоянии 1, поступит слово $\mathbf{x} = x_1 x_1 x_2 x_3 x_2 x_3$, то на выходе будет слово $\mathbf{y} = y_1 y_2 y_1 y_1 y_2 y_1$.

Теперь предположим, что автомат A является автоматом второго рода. Тогда таблица 4.10 определяет сдвинутую функцию выхода. При поступлении на вход автомата второго рода слова $\mathbf{x} = x_1x_1x_2x_3x_2x_3$, такого же, как и в случае автомата первого рода, на выходе появится слово $\mathbf{y} = y_2y_1y_1y_2y_1y_2$, которое отличается от соответствующего выходного слова автомата первого рода. Поэтому отображение, индуцируемое автоматом первого рода, отличается от отображения, индуцируемого автоматом второго рода.

Построим автомат A' первого рода, эквивалентный автомату A первого рода. Подставляя в сдвинутую функцию выхода $\lambda(q, x)$, заданную таблицей 4.10, функцию перехода $\delta(q, x)$ из табл. 4.11, получим обычную функцию выхода $\lambda'(q, x)$ (см. табл. 4.12).

Таблица 4.12

$q \backslash x$	x_1	x_2	x_3
1	y_2	y_1	y_2
2	y_1	y_2	y_1
3	y_1	y_1	y_1
4	y_2	y_1	y_2

Функция $\lambda'(q, x)$ задает автомат первого рода $A' = (X, Q, Y, q_1 \in Q, \delta, \lambda')$.

Объединяя таблицу выходов 4.12 и таблицу переходов 4.10, получим автоматную таблицу автомата A' первого рода, интерпретирующего автомат A второго рода:

$q \backslash x$	x_1	x_2	x_3
1	$2, y_2$	$4, y_1$	$1, y_2$
2	$1, y_1$	$3, y_2$	$4, y_1$
3	$1, y_1$	$4, y_1$	$2, y_1$
4	$4, y_2$	$1, y_1$	$3, y_2$

Легко заметить, что при поступлении на вход автомата A' первого рода того же слова $\mathbf{x} = x_1x_1x_2x_3x_2x_3$, на выходе получим слово $\mathbf{y} = y_2y_1y_1y_2y_1y_2$, такое же, как и в случае автомата A второго рода. Таким образом, автомат A' интерпретирует автомат A .

Несколько сложнее показать (на этом останавливаться не будем), что для любого автомата первого рода можно построить эквивалентный ему автомат второго рода.

Необходимо еще раз подчеркнуть, что автоматы первого и второго рода мы различаем, когда интерпретируем работу конечного абстрактного автомата некоторым реальным устройством.

В дальнейшем по умолчанию будем считать, что задан автомат первого рода, если не оговорено обратное.

4.2.5. Гомоморфизм, изоморфизм и эквивалентность автоматов

Ранее уже упоминались эквивалентные автоматы. В этом разделе будет дано строгое определение эквивалентности. Но прежде дадим понятия гомоморфизма и изоморфизма.

Пусть $S=(X_S, Q_S, Y_S, \delta_S, \lambda_S)$, и $T=(X_T, Q_T, Y_T, \delta_T, \lambda_T)$ – два автомата. Три отображения $f: X_S \rightarrow X_T$, $g: Q_S \rightarrow Q_T$, и $h: Y_S \rightarrow Y_T$ называются *гомоморфизмом* автомата S в автомат T , если для любых $x \in X_S$, $q \in Q_S$, и $y \in Y_S$ выполняются условия

$$\begin{aligned}\delta_T(g(q), f(x)) &= g(\delta_S(q, x)), \\ \lambda_T(g(q), f(x)) &= h(\lambda_S(q, x)).\end{aligned}\tag{4.2.3}$$

В этом случае автомат T называется гомоморфным автомату S . Если все три отображения сюръективны, то эта тройка называется *гомоморфизмом* S на T . Если, кроме того, эти отображения взаимно однозначны, то они называются *изоморфизмом* S на T . Автоматы, для которых существует изоморфизм, называются изоморфными. Этот факт обозначается так: $S \sim T$.

Понятно, что мощности соответствующих алфавитов у таких автоматов должны быть одинаковыми. Изоморфизм можно пояснить так: автоматы S и T изоморфны, если входы, выходы и состояния S можно переименовать так, что автоматная таблица S превратится в автоматную таблицу T . Изоморфизм соответствующих графов переходов является необходимым, но недостаточным условием изоморфизма автоматов. При гомоморфизме, кроме переименования, происходит еще и "склеивание" некоторых состояний S в одно состояние T .

Теперь пусть оба автомата S и T имеют одинаковые входные и выходные алфавиты. Состояние q автомата S и состояние r автомата T называют *неотличимыми*, если для любого входного слова $S(q, x) = T(r, x)$.

Автоматы S и T называют неотличимыми, если для любого состояния q автомата S найдется неотличимое от него состояние r автомата T и наоборот. Неотличимость автоматов означает, что любое автоматное отображение, реализуемое одним из них, может быть реализовано другим. Иначе говоря, их возможности по переработке входной информации в вы-

ходную совпадают. Отношение неотличимости между состояниями (и автоматами) рефлексивно, симметрично и транзитивно, а следовательно, является отношением эквивалентности. Это и явилось основанием называть неотличимость эквивалентностью и говорить об *эквивалентных состояниях*, или *эквивалентных автоматах*, имея в виду отношение неотличимости.

Переход от автомата S к эквивалентному ему автомату называется эквивалентным преобразованием автомата S . Существует много различных задач по эквивалентному преобразованию автоматов к автомату с заданными свойствами.

4.2.6. Минимизация автоматов

Среди задач по эквивалентному преобразованию автоматов наиболее изученной и интересной является задача о минимизации числа состояний автомата или, более коротко, задача минимизации автомата: среди автоматов, эквивалентных заданному, найти автомат с наименьшим числом состояний – так называемый минимальный автомат. При интерпретации автомата цифровым устройством это означает минимизацию числа элементов памяти такого устройства.

Теорема 4.2.3. Для любого автомата S существует минимальный автомат S_0 , единственный с точностью до изоморфизма. Если множество состояний S разбивается на l классов эквивалентности ($l \leq n$):

$C_1 = \{q_{11}, q_{12}, \dots, q_{1l_1}\}, \dots, C_l = \{q_{l1}, q_{l2}, \dots, q_{ll_l}\}$, то S_0 имеет l состояний.

Доказательство. Если q_{j1} и q_{j2} – состояния из одного класса эквивалентности C_j , то для любой входной буквы x состояния $\delta_S(q_{j1}, x)$ и $\delta_S(q_{j2}, x)$ находятся в одном классе эквивалентности (допустим в C_k).

Действительно, если это не так (т.е. состояния $\delta_S(q_{j1}, x)$ и $\delta_S(q_{j2}, x)$ не эквивалентны), то найдется слово x такое, что $\delta_S(\delta_S(q_{j1}, x), x) \neq \delta_S(\delta_S(q_{j2}, x), x)$.

Тогда, учитывая соотношение (4.1.2), будем иметь $S(q_{j1}, xx) \neq S(q_{j2}, xx)$, то есть, q_{j1} и q_{j2} не эквивалентны, что противоречит условию. Теперь определим автомат $S_0 = (X_S, Q_{S_0}, Y_S, \delta_{S_0}, \lambda_{S_0})$ следующим образом: $Q_{S_0} = \{C_1, C_2, \dots, C_l\}$; для любого C_i и любой входной буквы x $\delta_{S_0}(C_i, x) = C_j$, где C_j - класс эквивалентности, содержащий состояние $\delta_S(q_{ir}, x)$ ($q_{ir} \in C_i$ - любое состояние из класса C_i); $\lambda_{S_0}(C_i, x) = \lambda_S(q_{ir}, x)$.

Ясно, что автомат S_0 эквивалентен S и по построению не имеет эквивалентных состояний.

Покажем теперь, что автомат S_0 минимален. Предположим, что это не так и имеется эквивалентный автомату S_0 автомат S_0' с меньшим числом состояний. Тогда по определению неотличимости для каждого состояния S_0 найдется эквивалентное ему состояние S_0' , а поскольку в автомате S_0' состояний меньше, чем в S_0 , то каким-то двум состояниям S_0 окажется эквивалентным одно состояние S_0' . В силу транзитивности эти два состояния S_0 будут эквивалентны, а это противоречит отсутствию в S_0 эквивалентных состояний. Следовательно, S_0 минимален.

Осталось показать, что любой другой минимальный автомат S_0'' изоморфен S_0 . Действительно, раз S_0'' минимален, он имеет такое же число состояний, что и S_0 , то есть различным состояниям S_0 соответствуют различные же состояния S_0'' . Это соответствие и есть искомый изоморфизм. Что и требовалось доказать.

Только что доказанная теорема, к сожалению, не конструктивна, поскольку не дает метода нахождения классов эквивалентности. Само определение неотличимости также не дает такого метода, поскольку предполагает перебор по бесконечному множеству входных слов. Среди многочисленных алгоритмов минимизации наибольшее распространение получил алгоритм Мили, который и приводится ниже (он описан индуктивно).

Дан автомат $S=(X,Q,Y,\delta,\lambda)$ с n состояниями. На каждом шаге алгоритма строится некоторое разбиение Q на классы, причем разбиение на каждом последующем шаге получается расщеплением некоторых классов предыдущего шага.

Шаг 1. Два состояния q и q' относим в один класс C_{1j} , если и только если для любого $x \in X$ $\lambda(q,x)=\lambda(q',x)$.

Шаг $i+1$. Два состояния q и q' из одного класса $C_{i,j}$ относим в один класс $C_{i+1,j}$, если и только если для любого $x \in X$ состояния $\delta(q,x)=\delta(q',x)$ принадлежат одному и тому же классу $C_{i,l}$. Если $i+1$ -й шаг не меняет разбиения, то алгоритм заканчивает свою работу и полученное разбиение является разбиением на классы эквивалентных состояний, в противном случае применяем шаг $i+1$ к полученному разбиению.

Так как на каждом шаге число классов увеличивается (а всего их не более n), то приведенный алгоритм заканчивается не больше, чем за $(n-1)$ шагов. Нетрудно показать, что алгоритм действительно дает разбиение на классы эквивалентности.

Пример 4.8. Для автомата A с восемью состояниями и двумя выходными буквами, заданного табл. 4.13, алгоритм строит следующую последовательность разбиений:

1-й шаг: $\{1,5\}, \{2,3,8\}, \{4,6,7\},$

2-й шаг: $\{1,5\}, \{2\}, \{3,8\}, \{4,6,7\},$

3-й шаг: $\{1,5\}$, $\{2\}$, $\{3,8\}$, $\{4,7\}$, $\{6\}$,

4-й шаг: $\{1,5\}$, $\{2\}$, $\{3\}$, $\{8\}$, $\{4,7\}$, $\{6\}$.

Таблица 4.13

$q \backslash x$	x_1	x_2	x_3
1	4,1	2,2	5,1
2	5,2	1,1	4,2
3	3,2	5,1	4,2
4	5,1	8,2	4,2
5	7,1	2,2	1,1
6	1,1	2,2	4,2
7	5,1	8,2	7,2
8	3,2	5,1	6,2

Последнее разбиение является искомым, т.е. минимальный автомат имеет шесть состояний. Если найденные классы переобозначить, например, по порядку: $1=\{1,5\}$; $2=\{2\}$, $3=\{3\}$, $4=\{8\}$, $5=\{4,7\}$, $6=\{6\}$, то автоматная таблица минимального автомата будет следующей:

$q \backslash x$	x_1	x_2	x_3
1	5,1	2,2	1,1
2	1,2	1,1	5,2
3	3,2	1,1	5,2
4	3,2	1,1	6,2
5	1,1	4,2	5,2
6	1,1	2,2	5,2

4.2.7. Частичные автоматы и их свойства

Представим себе, что хотя бы одна из двух функций δ или λ является не полностью определенной, то есть для некоторых пар (состояние - вход) функция перехода или выхода не определена. Это отражается наличием прочерков в соответствующих местах автоматной таблицы или матрицы соединений. В графе переходов, где функция δ не определена, нарушено условие полноты. В таких случаях автомат называется *частичным*, или не полностью определенным. Для частичного автомата задание функций δ и λ нуждаются в уточнении. Удобно при этом пользоваться значком \cong : запись $A \cong B$ означает, что либо A и B одновременно не определены, либо определены и равны.

Функция перехода $\delta(q_i, \mathbf{x})$:

- 1) для каждой входной буквы x_j функция $\delta(q_i, x_j)$ задана автоматной таблицей,
- 2) если функция $\delta(q_i, \mathbf{x})$ определена, то
$$\delta(q_i, \mathbf{x}x_j) \cong \delta(\delta(q_i, \mathbf{x}), x_j);$$
- 3) если функция $\delta(q_i, \mathbf{x})$ не определена, то $\delta(q_i, \mathbf{x}x_j)$ не определена для всех x_j .

Выходная функция $\lambda(q_i, \mathbf{x})$:

$$\lambda(q_i, \mathbf{x}x_j) \cong \lambda(\delta(q_i, \mathbf{x}), x_j).$$

Автоматный оператор $S(q_i, x_j)$:

- 1) $S(q_i, x_j) = \lambda(q_i, x_j)$ (если функция $\lambda(q_i, x_j)$ не определена, то значение $S(q_i, x_j)$ – это прочерк);
- 2) $S(q_i, \mathbf{x}x_j) = S(q_i, \mathbf{x})\lambda(\delta(q_i, \mathbf{x}), x_j)$, если $\delta(q_i, \mathbf{x})$ определена. В случае, если не определена $\lambda(\delta(q_i, \mathbf{x}), x_j)$, справа от $S(q_i, \mathbf{x})$ ставится прочерк;

3) если не определена функция $\delta(q_i, x)$, то не определено и отображение $S(q_i, xx_j)$.

Отсюда видна неравноправность функций δ и λ : если δ не определена на слове x , то она не определена и на всех его продолжениях, а для функции λ это не обязательно. На графе это наглядно видно: если функция $\delta(q_i, x)$ не определена, то это значит, что не определен путь x из вершины q_i , поэтому не ясно, как его продолжить. Если же $\delta(q_i, x)$ определена, следовательно, определен путь x из вершины q_i , то, идя по этому пути, можно прочесть и выходное слово (возможно, с прочерками там, где функция выхода не определена). Входное слово x , для которого автоматное отображение $S(q_i, x)$ определено, называется допустимым для q_i . Таким образом, отображение, индуцируемое частичным автоматом, является не чем иным, как частичным отображением, областью определения которого является множество допустимых слов данного автомата.

Понятие неотличимости для частичных автоматов также нуждается в корректировке. Наиболее простое обобщение этого понятия следующее. Состояния q_i автомата S и r_j автомата T называются *псевдонеотличимыми*, если для любого слова x $S(q_i, x) \cong T(r_j, x)$, то есть если области определения операторов S и T совпадают и в этих областях q_i и r_j эквивалентны. Автоматы S и T псевдонеотличимы, если для любого состояния S найдется псевдонеотличимое состояние T и наоборот. Достоинство такого определения в том, что оно совпадает с обычным определением неотличимости для вполне определенных автоматов и, кроме того, является отношением эквивалентности. Недостаток же этого определения в том, что оно искусственно сужает рассматриваемые классы псевдонеотличимых автоматов, требуя совпадения областей определения сравниваемых состояний. То есть понятие псевдонеотличимости слишком слабое и не учитывает всех возможностей, скажем, минимизации автоматов.

Для частичных автоматов часто используются понятия эквивалентного и изоморфного продолжения (покрытия) автоматов. Для иллюстрации этих понятий рассмотрим автомат, заданный табл. 4.14

Таблица 4.14

$q \backslash x$	x_1	x_2	x_3
1	2,0	-	3,-
2	-	1,-	3,0
3	2,1	1,-	3,0

Возьмем состояние 2 и 3 (q_2 и q_3). Область определения для q_2 содержится в области определения для q_3 и, кроме того, в области определения для q_2 выполняется условие $S(q_2, x) = S(q_3, x)$ для любого слова x , так как при любой входной букве x $\lambda(q_2, x) = \lambda(q_3, x)$ и $\delta(q_2, x) = \delta(q_3, x)$. Поскольку область определения для q_3 включает в себя область определения для q_2 , то можно сказать, что возможности состояния q_3 больше, чем q_2 , на тех словах, на которых $S(q_2, x)$ не определено, а $S(q_3, x)$ определено. Если заменить теперь состояние q_2 на q_3 (просто вычеркнуть строку q_2 , а переходы в q_2 поменять на переходы в q_3), то получим автомат S' , который «делает больше, чем S ». Говорят, что автомат S' покрывает автомат S , или является продолжением автомата S , или автомат S' содержит (включает) автомат S . В этом случае S' есть эквивалентное продолжение, покрытие или надавтомат автомата S , а S – эквивалентное сужение или подавтомат автомата S . Это обозначается как $S \subseteq S'$ или $S' \supseteq S$.

Дадим более строгое определение покрытия. Состояние q_i автомата S покрывает (включает) состояние r_j автомата T (S и T могут совпадать), если для любого слова x из того, что $T(r_j, x)$ определено, следует, что $S(q_i, x)$ определено и $T(r_j, x) = S(q_i, x)$. Автомат S включает (покрывает)

автомат T , если для любого состояния T найдется покрывающее его состояние S . Таким образом, автомат $S = (X_S, Q_S, Y_S, \delta_S, \lambda_S)$ покрывает автомат $T = (X_T, Q_T, Y_T, \delta_T, \lambda_T)$, если $X_T \subseteq X_S$, $Y_T \subseteq Y_S$, а автоматное отображение $S(q_S, \mathbf{x}_S)$ ($q_S \in Q_S, \mathbf{x}_S \in X_S^*$) продолжает отображение $T(q_T, \mathbf{x}_T)$ ($q_T \in Q_T, \mathbf{x}_T \in X_T^*$) на множестве X_S^* .

Пусть теперь S, T и W – автоматы, удовлетворяющие условиям $S \sim T$ и $T \subseteq W$. Тогда автомат S является изоморфно вложенным в W . Можно также сказать, что W является изоморфным продолжением S , а автомат S является изоморфным сужением автомата W . Обозначается это так $S \subseteq W$.

Можно показать, что отношение покрытия (равно как и изоморфного вложения) автоматов обладает следующими свойствами:

- $A \subseteq A$ (рефлексивность);
- $(A \subseteq B) \& (B \subseteq A) \mapsto A = B$ (антисимметричность),
- $(A \subseteq B) \& (B \subseteq C) \mapsto A \subseteq C$ (транзитивность),

то есть являются отношениями нестрогого порядка.

Вернемся к табл. 4.14 и обратим теперь внимание на состояния 1 и 2. Они примечательны тем, что можно придумать состояние, покрывающее и q_1 , и q_2 . Например, это будет некоторое состояние 4: 2,0; 1, - ; 3,0. Такие состояния q_1 и q_2 называются совместимыми.

Состояния q_i автомата S и r_j автомата T (может быть $S=T$) *совместимы*, если существует состояние p_k (возможно, какого-то третьего автомата W), покрывающее и q_i и r_j . Автоматы S и T *совместимы*, если существует автомат W , включающий S и T . Можно дать определение совместимости автоматов, рассматривая автоматные отображения: состояния q_j и r_j совместимы, если для любого слова \mathbf{x} либо одно из отображений $S(q_i, \mathbf{x})$ и $T(r_j, \mathbf{x})$ не определено, либо выходные слова $S(q_i, \mathbf{x})$ и $T(r_j, \mathbf{x})$ (они могут со-

держат прочерки) непротиворечивы, т.е. не содержат на одинаковых местах разных букв.

Используя понятия совместимости и покрытия, можно предложить план минимизации частичных автоматов, аналогичный методу минимизации вполне определенных автоматов: находим совместимые состояния и заменяем их покрывающим состоянием. Однако здесь имеются некоторые трудности. Отношение совместимости в отличие от неотличимости и отношения включения нетранзитивно и не является отношением эквивалентности. Это означает, что классы совместимости могут пересекаться. Система классов совместимости будет *полной*, если $C_1 \cup C_2 \cup \dots \cup C_k = Q$ и *замкнутой*, если из того, что состояния q и q' находятся в одном классе совместимости, например в C_i ($q \in C_i, q' \in C_i$), следует, что состояния $\delta(q, x)$ и $\delta(q', x)$ также находятся в одном классе совместимости, например в C_j , всякий раз, когда соответствующие функции перехода определены.

Имеется теорема (Полла–Ангера), аналогичная теореме 4.2.3:

Теорема 4.2.4. Если для частичного автомата имеется полная и замкнутая система классов совместимости $C_1 \dots C_k$, то существует автомат S' , включающий S . Автомат $S' = (X_S, Q_S, Y_S, \delta_S, \lambda_S)$ строится следующим образом. Множество состояний равно $Q_S = \{C_1, C_2, \dots, C_k\}$. Для любого C_i и любой буквы $x \in X_S$ функция $\delta_{S'}(C_i, x) = C_j$, если для некоторых $q \in C_i$ $\delta(q, x) \in C_j$; $\delta_{S'}(C_i, x)$ не определена, если для всех $q \in C_i$ $\delta_S(q, x)$ не определена. Функция выхода $\lambda_{S'}(C_i, x) = u$, если для некоторых $q \in C_i$ $\lambda_S(q, x) = u$ и $\lambda_{S'}(C_i, x)$ не определена, если для всех $q \in C_i$ функция $\lambda_S(q, x)$ не определена. Нетрудно видеть, что состояние C_i автомата S' покрывает все состояния из класса совместимости C_i автомата S и, следовательно (ввиду полноты системы классов $\{C_i\}$), автомат S' покрывает автомат S . Что и требовалось доказать.

Если автомат S полностью определен, обе теоремы (4.2.3 и 4.2.4) совпадают.

Для минимизации частичных автоматов можно использовать и алгоритм Мили. При этом нужно сначала построить различные доопределения частичного автомата до полных автоматов (конечно, они будут покрывать исходный автомат), а затем минимизировать полученные полные автоматы по алгоритму Мили.

Реализация этого пути на практике сталкивается со значительными, порой непреодолимыми трудностями. По крайней мере, две из них заслуживают особого внимания.

1. Доопределить частичный автомат S можно различным образом. При этом получаются автоматы, скажем S_1, \dots, S_N , неэквивалентные между собой. Соответствующие минимальные автоматы S_{1_0}, \dots, S_{N_0} могут иметь различное число состояний и также неэквивалентны между собой, то есть их нельзя получить друг из друга эквивалентными преобразованиями. Поэтому результат минимизации будет сильно зависеть от того, насколько удачно мы доопределим исходный частичный автомат. Кроме того, полученный результат нельзя улучшить эквивалентными преобразованиями и необходимо весь путь проделывать заново: доопределять автомат по-другому, находить классы эквивалентных состояний и т.д. Число различных вариантов доопределения достаточно велико: нетрудно подсчитать, что если $|Q_S|=n$, $|Y_S|=k$, функция перехода δ_S не определена в p клетках автоматной таблицы, а функция выхода λ_S – в r клетках, то это число равно $n^p \cdot k^r$.

2. Даже полный перебор всех вариантов доопределения может не привести к минимальному автомату. Алгоритм Мили дает систему непересекающихся классов совместимости, но ведь эти классы могут пересекать-

ся. Из-за возможности пересечения классов совместимости число различных вариантов минимизации еще больше числа вариантов доопределения.

Пример 4.9. Простой пример иллюстрирует вышеизложенное. Рассмотрим автомат S , заданный табл. 4.15:

Таблица 4.15

$q \backslash x$	x_1	x_2
1	1,-	2,0
2	3,0	1,0
3	2,1	1,0

Есть два варианта его доопределения: либо $\lambda(1, x_1)=0$, либо $\lambda(1, x_1)=1$. Легко видеть, что ни в первом, ни во втором случае автомат не минимизируется, так как не имеет эквивалентных состояний. Это означает, что исходный автомат S не имеет нетривиальной системы замкнутых непесекающихся классов совместимости. Но для автомата S существует замкнутая система пересекающихся классов совместимости $C_1=\{1,2\}$ и $C_2=\{1,3\}$ и по теореме 4.2.4 имеется автомат S' с двумя состояниями (табл. 4.16), включающий автомат S .

Таблица 4.16

	x_1	x_2
C_1	$C_2, 0$	$C_1, 0$
C_2	$C_1, 1$	$C_1, 0$

Перечисленные трудности заставляют искать дополнительные методы построения системы классов совместимости, некоторые из них изложены в [5].

4.3. Распознавание множеств автоматами

4.3.1. Понятие события и постановка задачи представления событий автоматами

Пусть $X=\{x_1...x_m\}$ произвольный входной алфавит, а X^* - множество всех слов в этом алфавите. Тогда любое подмножество $E\subseteq X^*$ назовем *событием* в алфавите X . Конечно, можно было бы просто говорить "множество слов", но термин "событие" прижился в теории автоматов и стал общепринятым.

Для простоты изложения далее будем оперировать с автоматами без выходов.

Событие $E\subseteq X^*$ назовем *представимым* в автомате $S=(X,Q,\delta,q_1,F)$, если $\delta(q_1,x)\in F$ тогда и только тогда, когда $x\in E$. Всякому автомату при заданных q_1 и F однозначно соответствует представимое в нем событие: на графе автомата оно соответствует множеству путей, ведущих из q_1 в вершины, принадлежащие множеству заключительных состояний F . Событие называется представимым, если существует конечный автомат, в котором оно представимо. Синонимом этого понятия является: множество определимое или допустимое, или распознаваемое автоматом. Другими словами представимое в автомате событие можно назвать множеством, разрешимым автоматом.

Начальное состояние q_1 также может относиться к множеству заключительных состояний $q_1\in F$. В этом случае автомат, ничего не имея на входе, уже что-то представляет. Принято считать, что это «что-то» – пустое слово (пустой символ ϵ) и оно содержится в событии, представимым этим автоматом. Для произвольного слова x выполняется равенство

$ex=xe=x$, то есть пустое слово e играет роль единицы в свободной полугруппе слов входного алфавита, где ассоциативной бинарной операцией является *конкатенация* (приписывание одного слова к другому).

Не нужно путать пустое слово e с пустым событием (пустым множеством \emptyset). Автомат представляет пустое событие \emptyset , если ни одно из его заключительных состояний не достижимо из начального состояния.

С введением пустого символа e можно легко превратить любое алфавитное отображение в автоматное. Это делается с помощью стандартного приема.

Предположим, что f - произвольное частичное отображение множества слов X^* в множество Y^* . Введем в алфавит X и Y пустую букву e и образуем новые алфавиты $X' = X \cup \{e\}$, $Y' = Y \cup \{e\}$. Рассмотрим произвольное слово $\mathbf{p} \in X^*$, имеющее длину n , которое отображением f переводится в слово длины m : $\mathbf{r} = f(\mathbf{p})$, $\mathbf{r} \in Y^*$. Обозначим через \mathbf{p}' слово, образованное из \mathbf{p} приписыванием справа m раз буквы e , а через \mathbf{r}' слово в алфавите Y' , полученное из \mathbf{r} приписыванием слева n раз буквы e . В результате получаем одинаковую длину слов \mathbf{p}' и \mathbf{r}' , равную $m + n$. Повторив этот прием для каждого слова $\mathbf{p} \in X^*$, на котором определено отображение f , и полагая $\mathbf{r}' = f'(\mathbf{p}')$, получим новое частичное отображение f' множества X'^* в множество Y'^* .

Теперь доопределим отображение f' на всех начальных отрезках \mathbf{p}'_i любого слова $\mathbf{p}' \in X'^*$. Назовем это пополнением отображения f' и обозначим f'' . Смысл такой операции заключается в следующем. Если \mathbf{p}'_1 - начальный отрезок слова \mathbf{p}' из области определения отображения f' , то полагаем $f'(\mathbf{p}'_1)$ равным начальному отрезку слова $f'(\mathbf{p}')$, имеющему равную с отрезком \mathbf{p}'_1 длину. В результате пополнения отображения f' получим новое отображение f'' , область определения которого удовлетворяет

условию полноты. Можно показать, что полученное отображение f'' удовлетворяет условию однозначности и, следовательно, является автоматным.

Таким образом, частичное отображение f'' , построенное из произвольного алфавитного отображения f , удовлетворяет условиям автоматности и является частичным автоматным отображением.

Стандартный прием в силу своей общности не всегда приводит к экономному решению в смысле расходования дополнительных букв и состояний.

До сих пор мы рассматривали конечную последовательность букв входного алфавита, то есть слова конечной длины. Но можно говорить, что автомат распознает и бесконечную последовательность букв $x = x_{i_1}x_{i_2}\dots$, если он представляет множество $E = \{x_{i_1}, x_{i_1}x_{i_2}\dots\}$, составленное из всех начальных отрезков бесконечного слова x . Оказывается, что не все события представимы в автоматах. Об этом говорит следующая теорема.

Теорема 4.3.1. Существуют события, непредставимые в автоматах, именно: никакая непериодическая бесконечная последовательность не распознаваема конечным автоматом.

Доказательство. Первая часть теоремы должна быть очевидна: во-первых, из сопоставления мощностей соответствующих множеств (множество всех событий континуально, а множество конечных автоматов счетное), а во-вторых, потому, что существуют неразрешимые множества.

Вторая часть теоремы говорит о том, что могут быть разрешимые множества, но не представимые в автоматах. Пример такой последовательности, скажем, а алфавите $\{0,1\}$: 010110111011110...

Действительно, предположим, что некоторая непериодическая последовательность $x = x_{i_1}x_{i_2}\dots$ все же распознаваема автоматом S с n со-

стояниями. Тогда для любого ее начального отрезка $\mathbf{x}_k = x_{i_1}x_{i_2}\dots x_{i_k}$ будет верным соотношение $\delta(q_1, \mathbf{x}_k) = q_{i_k}$, где q_{i_k} – заключительное состояние.

В процессе переработки последовательности \mathbf{x} автомат проходит последовательность заключительных состояний $q_{i_1}, \dots, q_{i_k}, \dots$, а так как множество Q_S конечно, то какое-то состояние q_{i_k} встретится дважды: $q_{i_k} = q_{i_{k+n}}$. Это означает, что $\delta(q_{i_k}, x_{i_{k+1}}x_{i_{k+2}}\dots x_{i_{k+n}}) = q_{i_k}$ (все состояния, проходимые автоматом, заключительные). Поэтому, если на вход автомата в состоянии q_1 подать периодическую бесконечную последовательность $\mathbf{x}_1 = x_{i_1}\dots x_{i_k}(x_{i_{k+1}}x_{i_{k+2}}\dots x_{i_{k+n}})$, где в скобках – период такой последовательности, то автомат будет проходить последовательность заключительных состояний. Это означает, что все начальные отрезки слова \mathbf{x}_1 входят в событие, представимое в автомате и, следовательно, автомат не отличает \mathbf{x}_1 от \mathbf{x} , то есть не распознает \mathbf{x} вопреки предположению. Что и требовалось доказать.

Из теоремы 4.3.1 следует, что класс множеств, распознаваемых автоматом, есть лишь часть (собственное подмножество) класса разрешимых множеств. Отсюда и из теоремы Райса¹ вытекает, что свойство множества "быть представимым в конечном автомате" алгоритмически неразрешимо. Поэтому не имеет смысла описывать эти множества в терминах произвольных разрешимых множеств, и требуются какие-то другие, более слабые, средства такого описания.

¹ Теорема Райса гласит, что никакое нетривиальное свойство вычислимых функций (или разрешимых множеств) не является алгоритмически разрешимым, т.е. по описанию алгоритма, вычисляющего некоторую функцию (формирующему некоторое множество) невозможно установить ее (функции) свойства.

4.3.2. Регулярные события и алгебра Клини

Зададим три операции над событиями R и S в алфавите X .

1. *Объединением* (дизъюнкцией) событий R и S называется событие P , обозначаемое $R \cup S = P$, которое образуется обычным теоретико-множественным объединением множеств R и S .

2. *Конкатенацией* (умножением) событий R и S будет событие $U = R \cdot S$, состоящее из слов вида $\mathbf{u} = \mathbf{r} \cdot \mathbf{s}$, где $\mathbf{u} \in U$, $\mathbf{r} \in R$, $\mathbf{s} \in S$, то есть слова события U образуются приписыванием справа любого слова события S к любому слову события R (но не наоборот!).

3. *Итерацией* события R называется событие

$$R^* = e \cup R \cup RR \cup RRR \cup \dots \cup R^i \cup \dots = \bigcup_{i=0}^{\infty} R^i.$$

Одноэлементные события, т.е. события $\{x_i\}$, где $x_i \in X$, будем называть элементарными и обозначать буквами x_i . Событие e , образованное пустым словом e , состоит из одного слова нулевой длины и также относится к элементарным событиям. Событие назовем *регулярным*, если оно может быть получено из элементарных событий путем конечного применения перечисленных операций: объединения, умножения и итерации, которые также назовем регулярными.

Таким образом, мы определили алгебру регулярных событий $(R; \cup, *)$, несущим множеством которой является множество регулярных событий, а сигнатурой – две бинарные и одна унарная операция. Образующие этой алгебры (называемой еще алгеброй Клини) являются элементарные события. Каждый элемент этой алгебры (регулярное событие) может быть описан регулярным выражением в алфавите $X = \{x_1, x_2, \dots, x_m\}$, которое определяется рекурсивно следующим образом:

1) символы x_1, x_2, \dots, x_m , e и \emptyset являются регулярными выражениями;

2) если R и S – регулярные выражения, то таковыми являются $R \cup S$, $R \cdot S$ и R^* ;

3) никакое другое выражение не является регулярным, если оно не получено путем конечного числа применения правил 1 и 2.

Таким образом, регулярное выражение – это формула в алгебре событий. Регулярные выражения эквивалентны, если они описывают одно и то же регулярное событие.

Эквивалентные соотношения в алгебре регулярных событий вытекают из свойств операций \cup , \cdot , $*$. Если P , R , и S – регулярные события, то имеют место соотношения;

$$P \cup \emptyset = \emptyset \cup P = P;$$

$$P \cdot \emptyset = \emptyset \cdot P = \emptyset;$$

$$P \cdot e = e \cdot P = \emptyset;$$

$$\emptyset^* = e;$$

$$e^* = e;$$

$$\left. \begin{aligned} P \cup R &= R \cup P, \\ P \cdot P^* &= P^* \cdot P \end{aligned} \right\} \text{ - коммутативность объединения и итерации;}$$

$$\left. \begin{aligned} P \cup (R \cup S) &= (P \cup R) \cup S, \\ P \cdot (R \cdot S) &= (P \cdot R) \cdot S \end{aligned} \right\} \text{ - ассоциативность объединения и умножения;}$$

ножения;

$$\left. \begin{aligned} P \cdot (R \cup S) &= P \cdot R \cup P \cdot S, \\ (P \cup R) \cdot S &= P \cdot S \cup P \cdot S \end{aligned} \right\} \text{ - левая и правая дистрибутивность умножения относительно объединения;}$$

жения относительно объединения;

$$P^* = e \cup P \cdot P^* \text{ – разворачивание итерации;}$$

$$\left. \begin{aligned} P \cup P &= P, \\ (P^*)^* &= P^* \end{aligned} \right\} \text{ – идемпотентность объединения и итерации;}$$

$$P^* \cup P = P^* \text{ – дизъюнктивное поглощение итерации;}$$

$$P^* \cdot P^* = P^* \text{ – мультипликативное поглощение итерации.}$$

Из рассмотрения операций $\cup, \bullet, *$ вытекает, что все конечные события регулярны. Это следует из того, что любое слово события выражается произведением букв, а любое конечное событие – объединением образующих его слов.

Бесконечное регулярное событие может появиться только благодаря итерации и наоборот, если в регулярном выражении присутствует операция итерации, то оно описывает бесконечное событие (если только итерация не применяется к пустой букве e , так как $e^* = e$, т.е. конечно).

Пример 4.10. Регулярное выражение $U = (x_1 \cup x_2 \cup \dots \cup x_m)^*$ задает множество всех слов (включая пустое) в алфавите $X = \{x_1, x_2, \dots, x_m\}$. Такое событие называется *универсальным* событием (по аналогии с универсальным множеством).

Пример 4.11. Регулярное выражение $E = (a \cup c)^* b (a \cup c)^*$ задает событие в алфавите $\{a, b, c\}$, состоящее из всех слов, содержащих букву b только один раз.

Регулярные события тесно связаны с автоматами. Эта связь дается фундаментальной теоремой Клини.

Теорема 4.3.2. (Клини). Класс событий, представимых в конечных автоматах, совпадает с классом регулярных событий.

По сути, эта теорема состоит из двух теорем.

Теорема 4.3.2 а (теорема синтеза). Для любого регулярного события существует конечный автомат, представляющий это событие.

Теорема 4.3.2 б (теорема анализа). Всякое событие, представимое конечным автоматом, непременно регулярно.

Чтобы подойти к доказательству этих теорем, введем понятие *источника* (синонимы: переходный граф сигналов, сигнальный граф), под

которым будем понимать ориентированный граф, в котором выделены начальные и заключительные вершины, и на каждом ребре написана буква из алфавита X либо e (пустое ребро). Каждый источник H однозначно определяет некоторое событие E в алфавите X , порождаемое множеством путей из начальных вершин в заключительные вершины. В этом случае говорят, что источник H представляет событие E . Источники, представляющие одно и то же событие, называются эквивалентными. Частный случай источника – это автомат без выхода.

Для любого источника H можно построить эквивалентный источник H_0 с двумя полюсами (с одной начальной вершиной и одной заключительной). Для такого построения нужно в H_0 ввести новую вершину q_0 (единственная начальная вершина) и соединить ее пустыми ребрами с прежними начальными вершинами в H , а также новую вершину q_z (единственную заключительную) и соединить с ней все заключительные вершины в H пустыми ребрами. В остальном H_0 совпадает с H .

Теорема 4.3.3. Для любого регулярного события E существует двухполюсный источник, представляющий E .

Доказательство. Теорема доказывается индукцией по глубине построения формулы регулярного события. Элементарное событие представляется сигнальным графом с двумя вершинами – начальной и заключительной и ребром, соединяющим эти вершины. Это ребро взвешено буквой x_i или e (рис. 4.4, а).

Если построены двухполюсные источники: H_1 , представляющий регулярное событие E_1 , и H_2 , представляющий регулярное событие E_2 , с начальными q_{01} , q_{02} и заключительными q_{z1} и q_{z2} вершинами соответственно, то источник H с начальной вершиной q_0 и заключительной – q_z , представляющий регулярное событие E – результат регулярной операции над E_1 и E_2 , строится следующим образом:

1) объединение $E=E_1 \cup E_2$ будет изображено параллельным соединением H_1 и H_2 (рис. 4.4, б). Из вершины q_0 проводятся пустые ребра в q_{01} и q_{02} , а из q_{z1} и q_{z2} проводятся пустые ребра в вершину q_z ;

2) конкатенация $E=E_1 E_2$ строится последовательным соединением H_1 и H_2 (рис. 4.4, в). Из q_{z1} проводится пустое ребро в q_{02} ; вершина q_{01} объявляется начальной у H , вершина q_{z2} — заключительной;

3) итерация $E = (E_1)^*$ получается зацикливанием H_1 (рис. 4.4, г): из вершины q_{z1} проводится пустое ребро в q_{01} , и эта же вершина q_{01} объявляется начальной и заключительной в H .

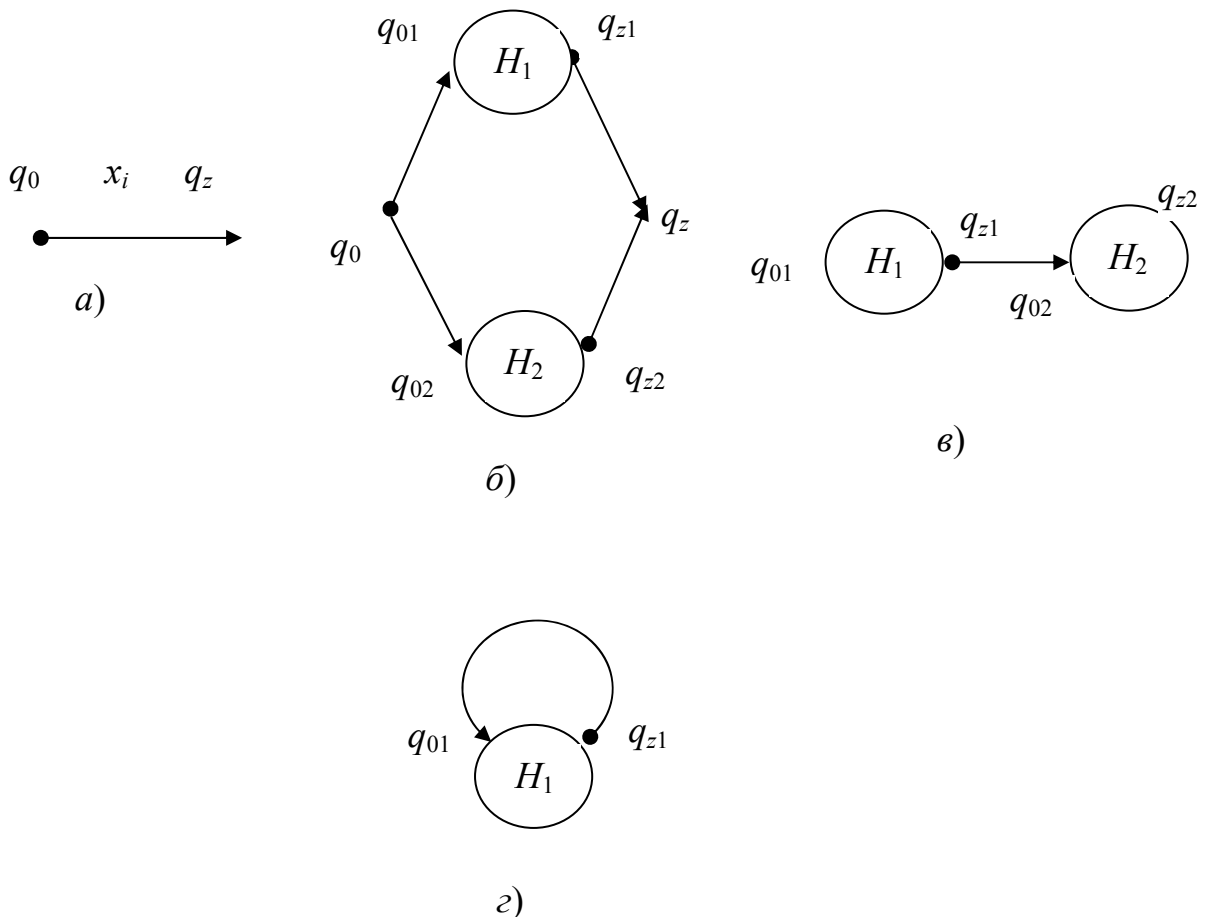


Рис. 4.4. Источники, представляющие регулярные операции

Построенные таким образом источники действительно представляют соответствующие события. Докажем это, например, для объединения

$E=E_1 \cup E_2$ (доказательства для умножения и итерации аналогичны). Возьмем $x \in E$. Тогда $x = x_1 \cup x_2$, где $x_1 \in E_1$, а $x_2 \in E_2$. По условию H_1 представляет E_1 , H_2 представляет E_2 , поэтому существует путь x_1 из q_{01} в q_{z1} и путь x_2 из q_{02} в q_{z2} . Тогда по построению существует путь $ex_1e \cup ex_2e = x_1 \cup x_2$ из вершины q_0 в вершину q_z . И наоборот, всякий путь x из q_0 в q_z обязательно проходит через q_{01} и q_{z1} либо через q_{02} и q_{z2} и имеет вид $x = x_1 \cup x_2$, где x_1 — путь из q_{01} в q_{z1} , а x_2 — путь из q_{02} в q_{z2} , откуда следует, что $x_1 \in E_1$ и $x_2 \in E_2$. Что и требовалось доказать.

При построении источника в некоторых случаях следует вводить пустые ребра. Это делается для того, чтобы избежать ложных путей. Система правил, когда следует вводить пустые ребра в граф регулярного выражения, следующая:

1. Пустые ребра вводятся в случае произведения двух или более итераций $S = \prod_{i \in N} (R_i)^*$ (рис. 4.5.), где R_i — регулярные выражения.

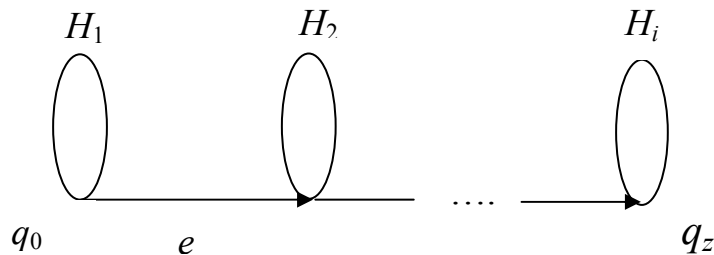


Рис. 4.5. Введение пустых ребер при произведении итераций

2. Пустые ребра в источнике, представляющем событие S , когда регулярное выражение для S начинается и заканчивается итерацией, вводятся в случаях:

- а) $S = (P^* \cdot R)^*$;
- б) $S = (R \cdot N^*)^*$;
- в) $S = (P^* \cdot R \cdot N^*)^*$.

Здесь R , P и N – произвольные регулярные выражения.

Соответствующие графы изображены на рис. 4.6.

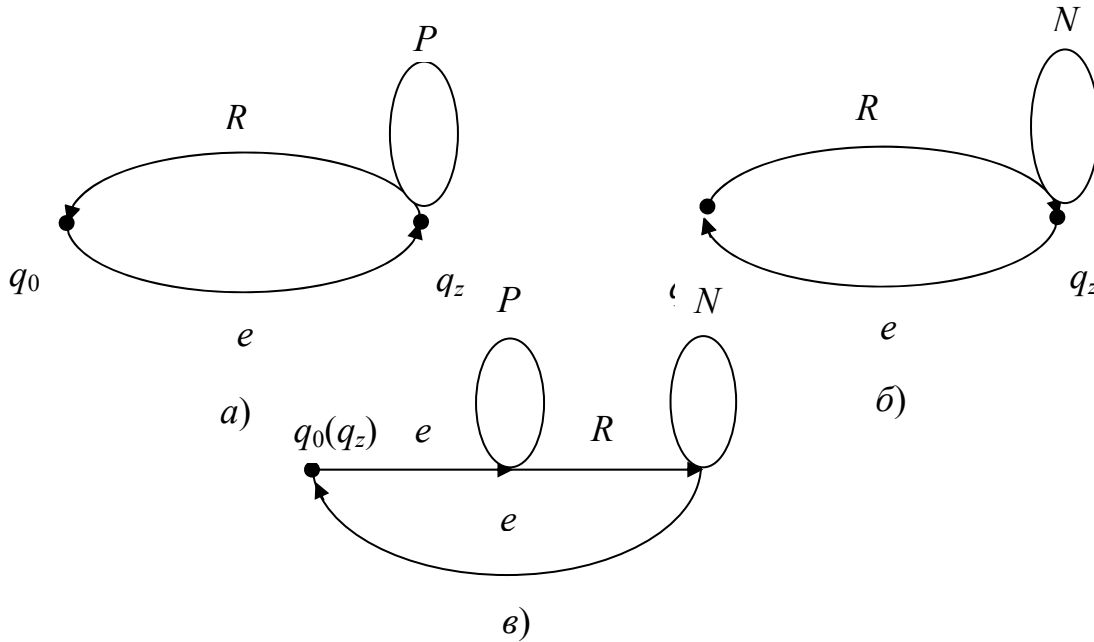


Рис 4.6. Введение пустых ребер при итерации

3. Пустые ребра вводятся в случае дизъюнкции, если хотя бы один из дизъюнктивных членов начинается с итерации

$$S = R^* \cdot Q \cup P^* \cup \dots \cup Q,$$

где Q – регулярное выражение, не содержащее итерации (рис. 4.7).

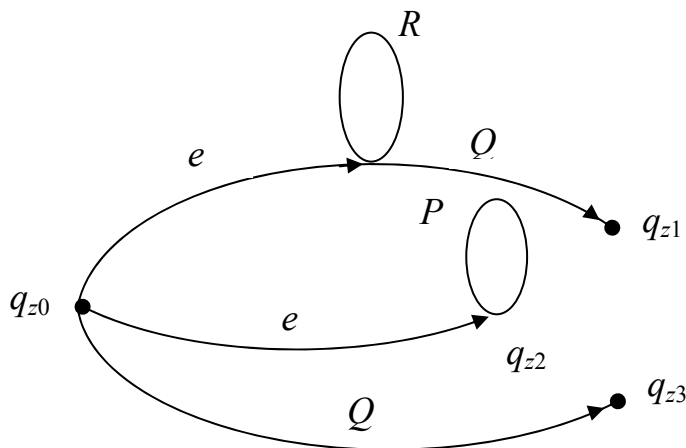


Рис. 4.7. Введение пустых ребер в случае дизъюнкции

4. Пустые ребра вводятся при умножении слева на дизъюнкцию, если хотя бы один из дизъюнктивных членов заканчивается итерацией (рис. 4.8.).

$$S = (Q \cdot R^* \cup P^* \cup \dots \cup Q) \cdot N.$$

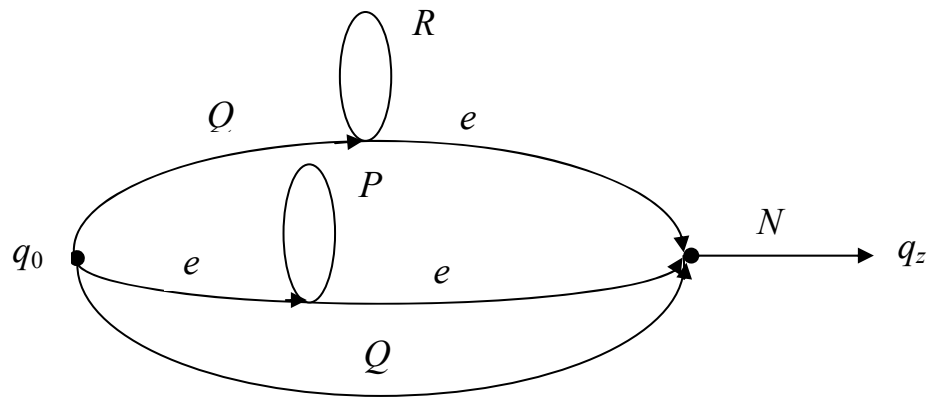


Рис. 4.8. Введение пустых ребер при умножении на дизъюнкцию

Перечисленная система правил является полной, т.е. ни в каких других случаях вводить пустые ребра нет необходимости. Это нетрудно показать, рассматривая всевозможные сочетания операций (\cup , \bullet , $*$) в регулярном выражении.

4.3.3. Синтез автоматов (абстрактный уровень)

Перейдем теперь к синтезу автоматов, т.е., по сути, к доказательству теоремы 4.3.2 а.

Вначале введем понятие *детерминированного источника*. Речь уже шла о том, что автомат без выходов – это частный случай источника. Источник будет являться графом автомата без выходов, если он: а) содержит одну начальную вершину; б) не содержит пустых ребер и в) удовлетворяет условиям автоматности. Такой источник и назовем детерминированным

источником в отличие от произвольного недетерминированного источника. Это название связано с тем, что произвольный источник можно интерпретировать как недетерминированный автомат, т. е. как автомат, в котором функция переходов $\delta(q, x)$ может иметь несколько значений (символу x при вершине q может соответствовать множество ребер, а слову x – множество путей из вершины q).

Теперь докажем вспомогательную теорему.

Теорема 4.3.4 (детерминизации). Для любого источника H с n вершинами существует эквивалентный ему детерминированный источник H' , имеющий не более чем 2^n вершин.

Доказательство. Назовем множество вершин \tilde{q} замкнутым, если из того, что $q_i \in \tilde{q}$, следует, что \tilde{q} принадлежит любая вершина, в которую из q_i ведет пустое ребро. Таким образом, для источника без пустых ребер любое множество вершин замкнуто.

Источник H' строится следующим образом. Образует все замкнутые подмножества вершин H (а их не более чем 2^n) и каждому такому подмножеству поставим в соответствие вершину \tilde{q}_i источника H' .

Через \tilde{q} обозначим наименьшее замкнутое подмножество вершин H , содержащее все начальные вершины H .

Это будет начальная вершина H' . Заключительными вершинами H' объявим все подмножества \tilde{q}_i , содержащее хотя бы одну заключительную вершину H . Если из множества \tilde{q}_i источника H есть пути x ($x \in X$) в множество \tilde{q}_j , то в источнике H' соединяем вершину \tilde{q}_i с вершиной \tilde{q}_j ребром, на котором написан символ x . Если же в H никакая из вершин \tilde{q}_i не имеет выходящего из нее ребра с символом x , то соединяем вершину \tilde{q}_i в H' с вершиной \emptyset (пустое подмножество вершин H) ребром x . Таким образом, каждой вершине \tilde{q}_i в H' и каждому символу x соответствует ровно одно

ребро x , выходящее из \tilde{q}_i , и источник H' является детерминированным. Построение ребер H' определяет функцию перехода автомата, граф которого – это источник H' . Начальное состояние этого автомата \tilde{q}_1 .

Осталось показать, что источник H' эквивалентен H . Действительно, H' обладает свойством: в H' непустой путь x из \tilde{q}_1 в \tilde{q}_j существует тогда и только тогда, когда в H для любой вершины $q \in \tilde{q}_j$ существует путь x из некоторой начальной вершины $q_1 \in \tilde{q}_1$ в q . Пустым путь x быть не может, так как, если $x=e$, то $\tilde{q}_j=\tilde{q}_1$ по условию замкнутости и в H' пустых ребер по построению нет. Это свойство доказывается индукцией по длине слова x .

Если $x=x$ (состоит из одного символа), то это свойство выполняется по построению ребер в H' . Предположим теперь, что оно выполняется и для слова длины, меньшей или равной k , и докажем, что оно выполняется и для слова xx , где $x \in X$ произвольная буква входного алфавита.

Пусть в H' имеется непустой путь xx из \tilde{q}_1 в \tilde{q}_j : $\delta(\tilde{q}_1, xx) = \tilde{q}_j$. Если $\delta(\tilde{q}_1, x) = \tilde{q}_k$, то из \tilde{q}_k в \tilde{q}_j ведет ребро x . По предположению, в H для любой вершины $q^* \in \tilde{q}_1$ существует путь x из начальной вершины q_1 в q^* . По построению H' из того, что в H' есть ребро x из \tilde{q}_k в \tilde{q}_j , следует, что в H для любой вершины $q \in \tilde{q}_j$ найдется вершина из подмножества \tilde{q}_k , из которой ведет путь x в q , поэтому в H имеется путь из q^* в q и, следовательно, путь xx из q_1 в q .

И наоборот, если в H для любой вершины $q \in \tilde{q}_j$ есть путь xx из начальной вершины $q_1 \in \tilde{q}_1$ в вершину q , то в H' будет выполняться условие $\delta(\tilde{q}_1, xx) = \tilde{q}_j$. Доказывается это аналогичным образом. При этом рассматривается множество путей xx из начальных вершин H в вершины множества \tilde{q}_j и множества \tilde{q}_k всех вершин, в которые ведут отрезки x этих путей.

Из доказанного свойства H' и определения заключительных вершин H' следует, что в H' путь x из \tilde{q}_1 в заключительную вершину есть тогда и только тогда, когда в H имеется путь x из некоторой начальной вершины в заключительную. Следовательно, источники H и H' эквивалентны, поскольку представляют одно и то же событие. Что и требовалось доказать.

По сути дела, из доказательства теорем 4.3.3 и 4.3.4 и следует доказательство теоремы синтеза 4.3.2 а, а синтез автомата, представляющего произвольное регулярное событие E , состоит в том, что сначала строится источник, представляющий E , а затем этот источник детерминируется согласно процедуре, изложенной при доказательстве теоремы 4.3.4.

Практически детерминизация упрощается в связи с тем, что некоторые подмножества вершин H (состояния H') не достижимы из начального состояния и их удаление не изменит события, представляемого автоматом. Поэтому в матрицу переходов H' включаются только те подмножества, которые порождаются процедурой детерминизации, начатой с подмножества \tilde{q}_1 . В этом случае построенный автомат может иметь меньше чем 2^n состояний.

Пример 4.12. Рассмотрим синтез автомата на абстрактном уровне. Регулярное событие, которое должен представлять автомат, может быть задано либо словесным описанием, либо регулярным выражением. Пусть регулярное событие в алфавите $\{a, b, c\}$ задано выражением

$$(a^* b \cup c \cdot c^*) \cdot (a \cup b)^* \cdot (ac)^*. \quad (4.3.1)$$

Процедура синтеза начинается с построения источника H . При этом нужно учитывать правила введения пустых ребер. В формуле (4.3.1) имеется произведение итераций (правило 1); первая скобка представляет собой дизъюнкцию, в которой один из дизъюнктивных членов начинается с итерации (правило 3), а также один из дизъюнктивных членов заканчивается

итерацией (правило 4). Поэтому построенный источник H будет содержать пустые ребра (см. рис. 4.9).

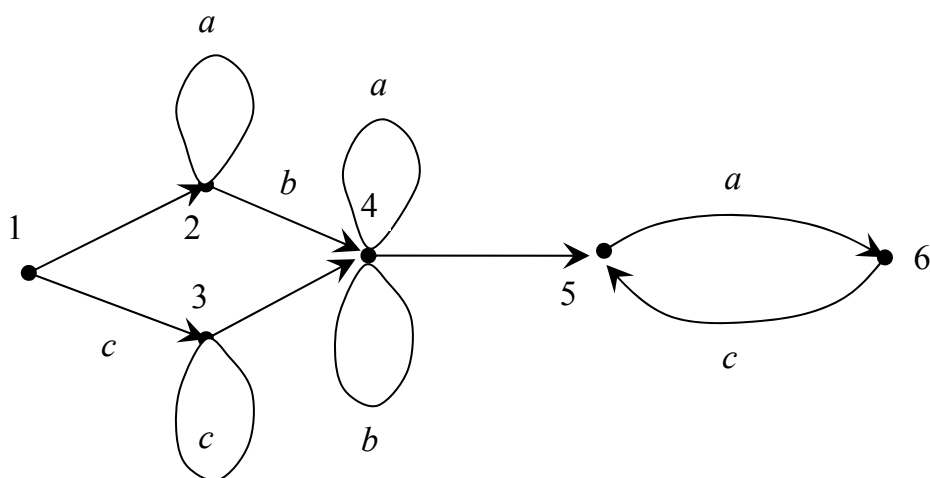


Рис. 4.9. Источник, представляющий событие (4.3.1)

На рис. 4.9 ребра, которым не приписано букв, – пустые. Начальная вершина источника – 1, заключительная – 5.

Следующий этап – детерминизация источника H в соответствии с теоремой 4.3.4. При этом строим только замкнутые подмножества, достижимые из начального замкнутого подмножества $\{1,2\}$. Функция переходов полученного автомата приведена в табл. 4.17.

Таблица 4.17

	a	b	c
$\{1,2\}$	$\{2\}$	$\{4,5\}$	$\{3,4,5\}$
$\{2\}$	$\{2\}$	$\{4,5\}$	\emptyset
$\{4,5\}$	$\{4,5,6\}$	$\{4,5\}$	\emptyset
$\{3,4,5\}$	$\{4,5,6\}$	$\{4,5\}$	$\{3,4,5\}$
\emptyset	\emptyset	\emptyset	\emptyset
$\{4,5,6\}$	$\{4,5,6\}$	$\{4,5\}$	$\{5\}$
$\{5\}$	$\{6\}$	\emptyset	\emptyset
$\{6\}$	\emptyset	\emptyset	$\{5\}$

В этой таблице знаком \emptyset обозначено пустое множество. После переобозначения подмножеств таблица переходов автомата приобретает следующий вид (табл. 4.18).

Таблица 4.18

	<i>a</i>	<i>b</i>	<i>c</i>
1	2	3	4
2	2	3	5
3	5	3	5
4	5	3	4
5	5	5	5
6	5	3	7
7	8	5	5
8	5	5	7

В табл. 4.18 выделены жирным шрифтом заключительные состояния 3,4,6 и 7, соответствующие подмножествам из табл. 4.17, содержащие заключительное состояние 5 источника *H*.

4.3.4. Анализ автоматов (абстрактный уровень)

Для процедуры анализа автоматов потребуется ввести несколько новых понятий. Ребра и вершины источника, не входящие в контур обратных связей, назовем *каскадными*. Вершины называются *стоком*, если они имеют только входящие ребра и *истоком*, если они имеют только выходящие ребра. Две вершины, лежащие в контуре обратной связи, называются *спаренными*.

Источник, состоящий только из каскадных вершин, называется *каскадным*. Поскольку стоки и истоки всегда каскадные, то любую из вершин обратной связи можно сделать каскадной с помощью операции, называемой *расщеплением* вершины. Произвольная вершина *q* в этом случае расщепляется на две вершины: *q'*, которая называется истоком, и *q''*, служащая стоком. Пример такого расщепления приведен на рис. 4.10.

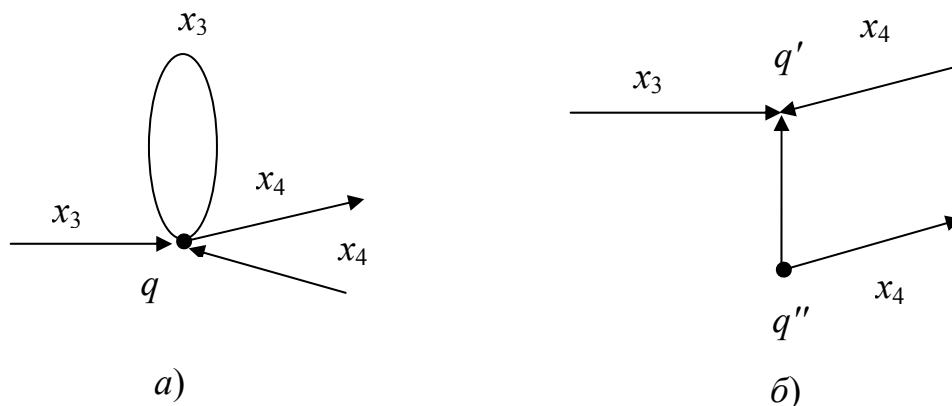


Рис. 4.10. Пример расщепления вершины

Расщепленные вершины называются индексными вершинами. Минимальное число вершин, которые нужно расщепить, чтобы разбить все контуры обратной связи, называется *индексом соединения* обратной связи.

Граф, изображающий источник, можно упростить, устранив ряд вершин и приведя ветевой переход к значению пути через устраненные вершины. Полученный граф называется *остатком* первоначального. Вершина, входящая в остаточный граф, называется остаточной. Путь, если он связывает остаточную вершину с собой или с заданной остаточной вершиной и не проходит через другие остаточные вершины, называется остаточным.

Исключая в исходном источнике все вершины, кроме истоков, стоков и индексных вершин, получим *индексный остаток*. Если необходимо сохранить вершину q , не являющуюся ни истоком, ни стоком, ни индексной вершиной, то это можно сделать путем соединения новой вершины q' с вершиной q ребром e и устранением вершины q . В результате получается исток. Аналогичным образом поступают и для образования стока.

Существующие методы анализа можно разделить на графические, использующие понятие источника и его эквивалентные преобразования, и аналитические, в которых применяются уравнения в алгебре регулярных событий. Впервые алгоритм получения регулярных выражений был дан

Мак-Нотоном и Ямадой (Mc. Naughton & Yamada) в 1960 г., однако он довольно громоздок и не приводится.

Рассмотрим графический алгоритм анализа. Отыскание регулярного выражения, означающего множество слов, переводящих автомат из состояния q_i в состояние q_j , сводится в конечном итоге к нахождению ветвевых переходов из вершины q_i в q_j на графе автомата. В этом случае граф автомата приводится к источнику, имеющему только два состояния: q_i и q_j . Если в начале приведения вершина q_i является истоком, q_j – стоком, то полученный источник будет иметь только один непустой переход от q_i к q_j и вес этого перехода как раз и будет являться искомым регулярным выражением. При таком приведении остальные вершины должны быть удалены. Например, пусть необходимо удалить в графе вершину q_k с петлей t_{kk} (см. рис. 4.11, а).

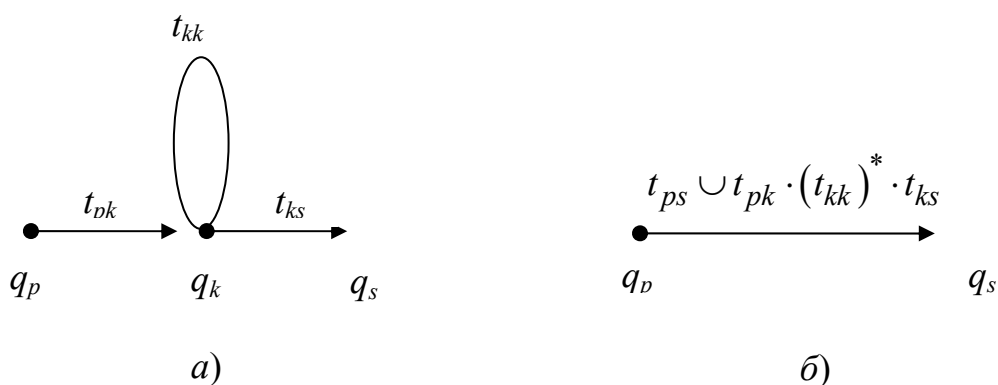


Рис.4.11.Удаление вершины с петлей

Регулярное выражение, связывающее q_p и q_s , будет $R_1 = t_{pk} \cdot (t_{kk})^* \cdot t_{ks}$, а при устранении вершины q_k получим (рис. 4.11,б): $R = t_{ps} \cup t_{pk} \cdot (t_{kk})^* \cdot t_{ks}$.

Таким образом, любой граф автомата можно свести к источнику с двумя вершинами q_i и q_j , ветвевым переходом между которыми и есть искомым регулярным выражением, представимое состоянием q_j при условии, что q_i – начальное состояние.

При анализе граф автомата, как правило, приводят к индексному остатку. Это можно сделать с помощью элементарных преобразований, но на практике используют приведение к индексному остатку с проверкой по правилу. Это правило заключается в том, что между остаточными вершинами q_i и q_j в построенном источнике должно быть ребро, если в первоначальном графе есть, по крайней мере, один остаточный путь от q_i к q_j . Вес такого ребра равен объединению всех приращений остаточных путей от q_i к q_j . Прирост пути от q_i к q_j определяется произведением приращений ребер, образующих этот путь.

Если индексный остаток включает сложный контур обратной связи, устраняемые вершины могут быть исключены расщеплением вершин. При удалении некоторой вершины q_k все другие вершины расщепляются на истоки и стоки. Далее вычисляются пути от истока до стока, и расщепленные вершины соединяются вновь.

Пример 4.13. Исходный граф автомата изображен на рисунке 4.12,а. Пусть q_1 – начальное состояние, а q_2 – заключительное. Расщепляем вершину q_1 (рис. 4.12, б). Вычисляем переход от q_1' к q_1'' (рис. 4.12, в). Соединяем q_1' и q_1'' (рис. 4.12, г). Записываем окончательный результат (рис. 4.12, д).

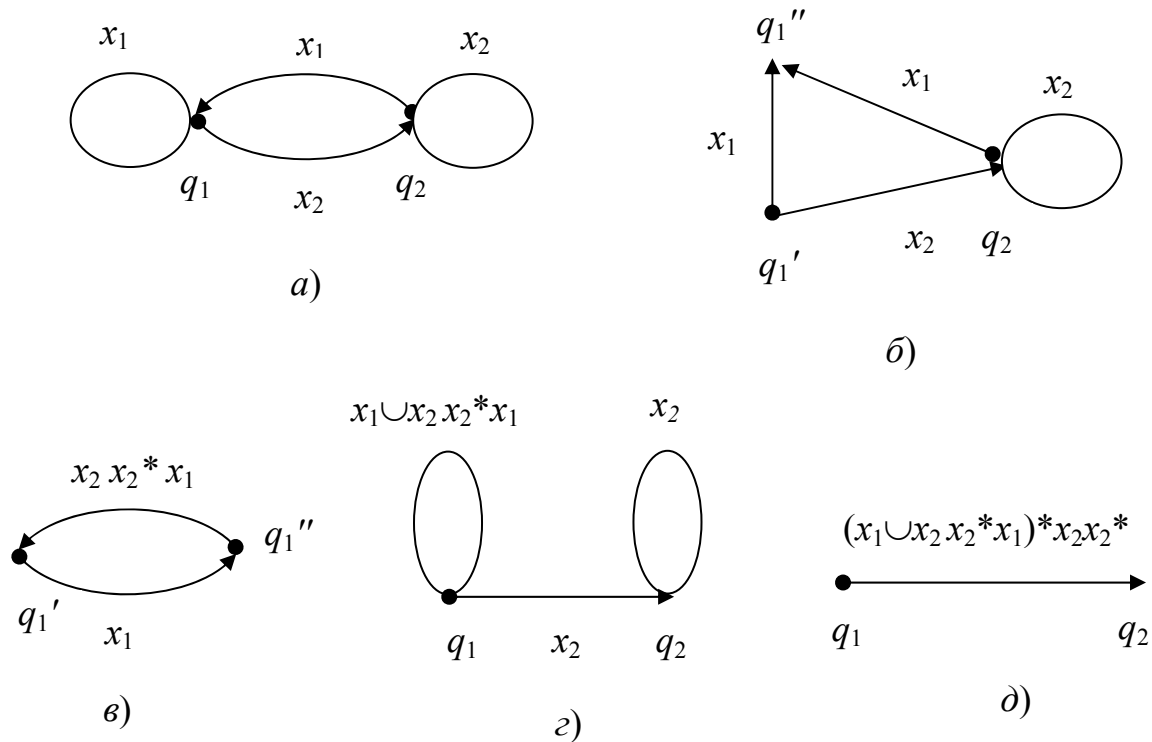


Рис. 4.12. Пример расщепления сложного контура обратной связи

Теперь можно сформулировать графический алгоритм анализа абстрактных автоматов.

1. По графу автомата находим исток и сток. Если их нет, то с начальной вершиной q_i соединяется пустым ребром новая вершина q_i' , а заключительная вершина q_j соединяется пустым ребром с новой вершиной q_j' . После этого q_i' берется как исток, а q_j' — как сток.

2. Все параллельные пути приводятся к форме $x_k \cup x_s$, а все последовательные — к форме $x_k x_s$.

3. Получаем индексный остаток графа, отмечая вершины q_i , q_j и индексные вершины. Находим приращения остаточных дуг.

4. Устраняем последовательно все индексные вершины индексного остатка графа. Приращение пути от вершины q_i к вершине q_j есть регулярное выражение события, представимого в автомате состоянием q_j при условии, что q_i — начальное состояние автомата.

Пример 4.14. Рассмотрим автомат, заданный своим графом переходов (см. рис. 4.13).

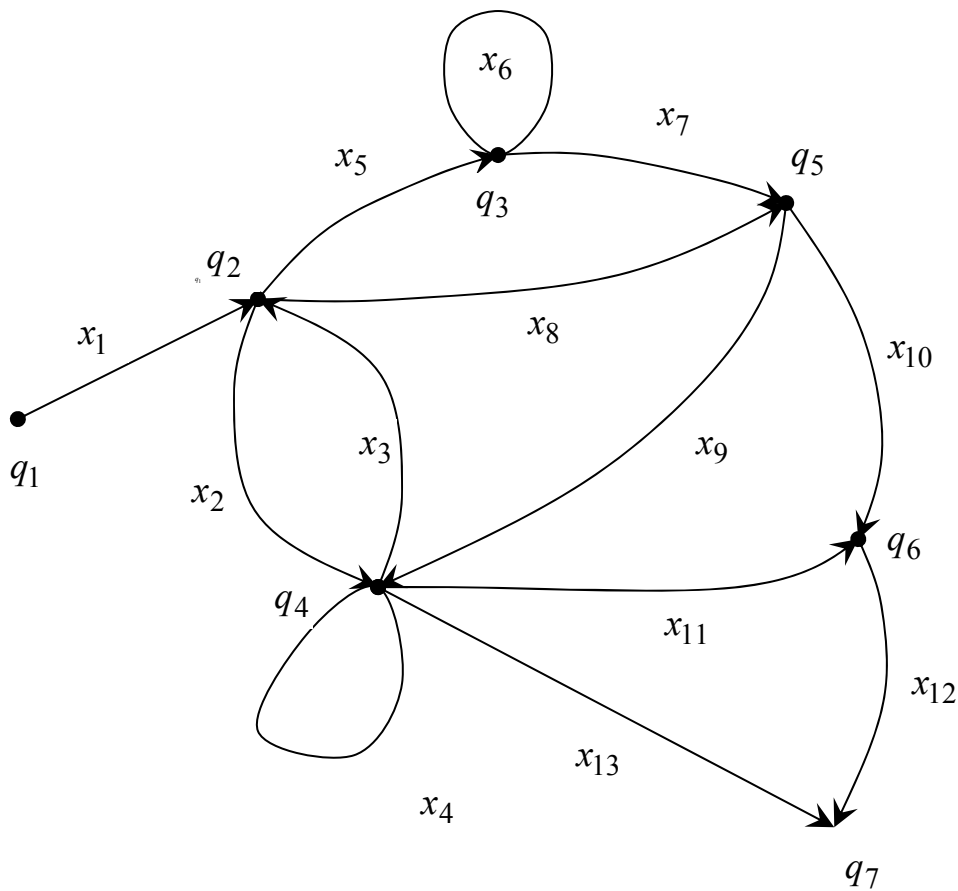


Рис. 4.13. Граф автомата

Пусть начальное состояние автомата q_1 , а заключительное — q_7 . Требуется провести анализ автомата, т.е. найти регулярное выражение, представимое этим автоматом. Поскольку вершина q_1 является истоком, а вершина q_7 — стоком, вводить дополнительные вершины не требуется. В графе на рис. 4.13 две индексные вершины — q_2 и q_4 . Можно оставить любую из них, например, q_4 . Тогда индексный остаток графа будет таким, как на рис. 4.14.

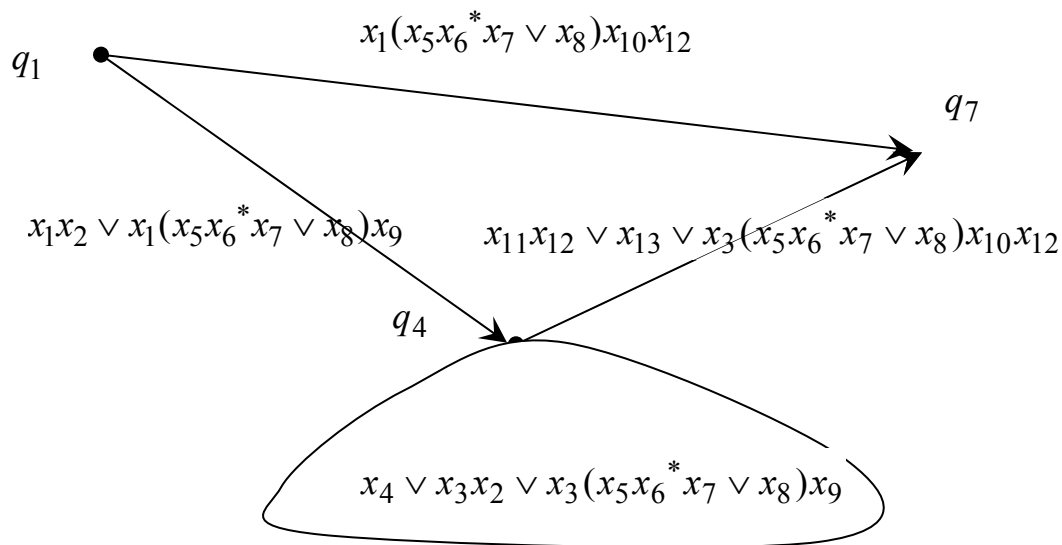


Рис. 4.14. Индексный остаток графа

Путь $x_3(x_5x_6^*x_7 \vee x_8)x_9(x_{13} \vee x_{11}x_{12})$ от вершины q_4 до вершины q_7 не является остаточным, так как проходит через индексную вершину q_4 . Устраняя индексную вершину q_4 , получаем окончательное выражение для регулярного события, представимого автоматом при условии, что начальное состояние автомата – q_1 , а заключительное – q_4 :

$$(x_1x_2 \vee x_1(x_5x_6^*x_7 \vee x_8)x_9)(x_4 \vee x_3x_2 \vee x_3(x_5x_6^*x_7 \vee x_8)x_9)^*(x_{11}x_{12} \vee x_{13} \vee x_3(x_5x_6^*x_7 \vee x_8)x_{10}x_{12}) \vee x_1(x_5x_6^*x_7 \vee x_8)x_{10}x_{12}.$$

4.4. Алгебра абстрактных автоматов

Рассмотрим свойства теоретико-множественных и алгебраических операций на множестве абстрактных автоматов. Есть две теоретико-множественные операции – объединение и пересечение, а также четыре алгебраические – умножение, суммирование, суперпозиция и композиция. Все эти операции бинарные и играют важную роль при синтезе автоматов,

так как на структурном уровне они соответствуют различным способам соединения простых (в частности элементарных) автоматов между собой при построении структурных схем более сложных автоматов.

Множество автоматов совместно с операциями над ними образуют алгебру абстрактных автоматов, которую не нужно путать с алгеброй регулярных событий на множестве слов входного алфавита произвольного автомата.

Задать определенную бинарную операцию на множестве автоматов означает указать закон, по которому любым двум автоматам из некоторого множества автоматов сопоставляется третий автомат из этого же множества. Какое именно множество имеется в виду, зависит от конкретного случая (от конкретной операции), а равенство понимается, как правило, с точностью до изоморфизма.

4.4.1. Теоретико-множественные операции

Операции объединения и пересечения автоматов играют вспомогательную роль при задании алгебраических операций, хотя их можно рассматривать и как самостоятельные операции на множестве $B(L)$ подавтоматов произвольного непустого автомата L . Тогда объединение двух автоматов $A \in B(L)$ и $B \in B(L)$ представляет автомат $C = A \cup B$, который является эквивалентным продолжением автоматов A и B , а пересечение представляет автомат $D \in B(L)$, по отношению к которому автоматы A и B являются эквивалентными продолжениями.

Зададим операции объединения и пересечения более конкретно. Пусть L – произвольный непустой автомат Мили, а $B(L)$ – множество его подавтоматов. Пусть далее $A \in B(L)$ и $B \in B(L)$ – подавтоматы автомата L , причем и A , и B имеют одинаковые начальные состояния, совпадающие с начальным состоянием L . Пусть заданы автомат $A = (X_1, Q_1, Y_1, q_1 \in Q_1,$

$F_1(x \in X_1/y \in Y_1))$ и автомат $B=(X_2, Q_2, Y_2, q_1 \in Q_2, F_2(x \in X_2/y \in Y_2))$. Тогда автомат $C=(X, Q, Y, q \in Q, F(x \in X/y \in Y))$ будет являться объединением A и B , если множества X , Y , Q и отображение F определяются по формулам

$$X=(\{1\} \times X_1) \cup (\{2\} \times X_2); \quad (4.4.1)$$

$$Q=Q_1 \cup Q_2; \quad (4.4.2)$$

$$Y=(\{1\} \times Y_1) \cup (\{2\} \times Y_2); \quad (4.4.3)$$

$$Fq=F_1q \cup F_2q, \quad (4.4.4)$$

где $q \in Q$. Для тех состояний, когда $q \notin Q_1$, полагаем $F_1q=\emptyset$, а при $q \notin Q_2$ имеем $F_2q=\emptyset$.

Если не происходит нарушения автоматности для автомата C , то есть равенство

$$F_1q(x/y)=F_2q(x/y)$$

не нарушается для любых $q \in Q$, $x \in X$ и $y \in Y$ (когда оба отображения не пусты) или если

$$X_1 \cap X_2 = \emptyset; \quad (4.4.5)$$

$$Y_1 \cap Y_2 = \emptyset, \quad (4.4.6)$$

то формулы (4.4.1) и (4.4.3) упрощаются и принимают вид

$$X=X_1 \cup X_2; \quad (4.4.7)$$

$$Y=Y_1 \cup Y_2. \quad (4.4.8)$$

В том случае, когда A и B вполне определенные автоматы, автомат $C=A \cup B$ также вполне определен, в противном случае автомат C будет частичным.

Пример 4.15. Автоматы заданы своими автоматными таблицами.

A		
$q \backslash x$	x_1	x_2
1	-	$3, y_2$
2	$2, y_2$	-
3	$3, y_1$	$2, y_3$

B			
$q \backslash x$	x_1	x_2	x_3
1	$2, y_1$	$3, y_2$	-
2	$2, y_2$	-	$1, y_1$
3	-	-	$1, y_3$

Найдем их объединение $C = A \cup B$. Поскольку нарушения условий автоматности не происходит, то нет необходимости различать одинаковые буквы алфавитов, и поэтому пользуемся формулами (4.4.7) (4.4.8), а также формулами (4.4.2) и (4.4.4). В результате получаем автомат C :

C

$q \backslash x$	x_1	x_2	x_3
1	$2, y_1$	$3, y_2$	-
2	$2, y_2$	-	$1, y_1$
3	$3, y_1 -$	$2, y_3 -$	$1, y_3$

Операцию объединения с соответствующими формулами (4.4.1) – (4.4.8) легко объединить и на случай n автоматов.

Из формул (4.4.2), (4.4.4), (4.4.5) – (4.4.8) следует, что каждый автомат Мили может быть представлен объединением автономных автоматов по входным и выходным буквам:

$$A = \left(\bigcup_{x \in X} A_x \right) \cup \left(\bigcup_{y \in Y} A_y \right).$$

Такое представление используется при разложении автоматов по различным операциям.

Автоматное отображение $S_C(q_1, \mathbf{x})$, индуцируемое автоматом C , есть продолжение автоматных отображений $S_A(q_1, \mathbf{x})$ на множество X^* .

Пересечением автоматов A и B будет являться автомат $D = A \cap B$, если его алфавиты X, Q, Y и отображение F определяется формулами

$$X = X_1 \cap X_2; \quad (4.4.9)$$

$$Q = Q_1 \cap Q_2; \quad (4.4.10)$$

$$Y = Y_1 \cap Y_2; \quad (4.4.11)$$

$$Fq = F_1q \cap F_2q, \quad (4.4.12)$$

где $q \in Q$.

Пример 4.16. Найдем пересечение автоматов A и B из примера 4.15.

Применяя формулы (4.4.9) – (4.4.12), получаем автоматную таблицу автомата $D = A \cap B$:

D

$q \backslash x$	x_1	x_2
1	-	$3, y_2$
2	$2, y_2$	-
3	-	-

Так же, как и операцию объединения, операцию пересечения, определяемую формулами (4.4.9) – (4.4.12), можно распространить на случай n автоматов.

Задание объединения и пересечения автоматов Мура наталкивается на трудности, связанные с тем, что одинаковые состояния могут иметь разные значения функции отметок, в связи с чем рекомендуется сначала интерпретировать автоматы Мура эквивалентными автоматами Мили, а затем находить их объединение или пересечение по известным формулам.

Нетрудно заметить, что объединение и пересечение автоматов ассоциативны, коммутативны и дистрибутивны.

4.4.2. Алгебраические операции

К алгебраическим операциям над автоматами относятся умножение, суммирование, суперпозиция и композиция.

Операция умножения графов приводит к двум операциям умножения автоматов. Первая операция, обозначаемая \times , применяется к произвольным автоматам с отдельными входами, то есть с разными входными алфавитами, а вторая обозначается \otimes и применяется к автоматам с общим входом, то есть с одним и тем же входным алфавитом.

Произведением произвольных непустых автоматов $A=(X,Q,Y,q_1\in Q,F(x\in X/y\in Y))$ и $B=(U,W,V,w_1\in W,P(u\in U/v\in V))$ будет называться автомат $K=(Z,H,S,h_1\in H,R(z\in Z/s\in S))$, у которого

$$Z=X\times U; \quad (4.4.13)$$

$$H=Q\times W; \quad (4.4.14)$$

$$S=Y\times V; \quad (4.4.15)$$

$$Rh=Fq\times Pw, \quad (4.4.16)$$

где $q\in Q$, $w\in W$, $h\in H$, $h=(q,w)$, $z=(x,u)$, $s=(y,v)$.

Начальным состоянием автомата $K=A\times B$ будет состояние $h_1=(q_1,w_1)$. Если оба автомата A и B вполне определенные, то и их произведение является вполне определенным автоматом. Если хотя бы один из исходных автоматов частичный, то в результате умножения получаем частичный автомат.

Можно определить операцию умножения и через матрицы соединений. Пусть имеется матрица соединений автомата A :

$$\mathbf{R}_A = \|r_{ij}(x/y)\|,$$

где $i, j \in \{1, 2, \dots, m\}$ и

$$r_{ij}(x/y) = \begin{cases} x/y, & \text{если } q_j \in F_{q_i} \text{ по букве } x \in X \text{ с выходом } y \in Y, \\ 0, & \text{если } q_j \notin F_{q_i}; \end{cases}$$

и матрица соединений автомата B :

$$\mathbf{R}_B = \|r_{kl}(u/w)\|,$$

где $k, l \in \{1, 2, \dots, n\}$ и

$$r_{kl}(u/v) = \begin{cases} u/v, & \text{если } w_l \in P_{w_k} \text{ по букве } u \in U \text{ с выходом } v \in V, \\ 0, & \text{если } w_l \notin P_{w_k}; \end{cases}$$

Матрица соединений \mathbf{R}_K автомата $K=A\times B$ равна прямому произведению матриц \mathbf{R}_A и \mathbf{R}_B , то есть

$$\mathbf{R}_K = \mathbf{R}_A \times \mathbf{R}_B,$$

а ее элементы определяются так:

$$r_{\alpha\beta}(z/s) = \begin{cases} (x,u)/(y,v), & \text{если } r_{ij} = x/y \text{ и } r_{kl} = u/v, \\ 0, & \text{если } r_{ij} = 0 \text{ или } r_{kl} = 0, \end{cases}$$

где $\alpha, \beta \in \{1, 2, \dots, p\}$, $p = m \cdot n$, $z = (x, u)$, $s = (y, v)$.

Пример 4.17. Автоматы A и B заданы автоматными таблицами:

A			B		
$q \backslash x$	x_1	x_2	$w \backslash u$	u_1	u_2
q_1	q_2, y_1	q_1, y_2	w_1	w_1, v_2	-
q_2	q_3, y_2	-	w_2	w_1, v_1	w_2, v_1
q_3	-	q_3, y_1			

Найдем произведение этих автоматов, т.е. автомат $K = A \times B$.

Входной алфавит автомата K – это упорядоченные пары входных букв автоматов A и B : обозначим их $Z = \{z_1, z_2, z_3, z_4\}$,

где $z_1 = (x_1, u_1)$, $z_2 = (x_1, u_2)$, $z_3 = (x_2, u_1)$, $z_4 = (x_2, u_2)$.

Выходной алфавит автомата K обозначим через $S = \{s_1, s_2, s_3, s_4\}$,

где $s_1 = (y_1, v_1)$, $s_2 = (y_1, v_2)$, $s_3 = (y_2, v_1)$, $s_4 = (y_2, v_2)$.

Алфавит состояний автомата K – это $H = \{h_1, h_2, h_3, h_4, h_5, h_6\}$,

где $h_1 = (q_1, w_1)$, $h_2 = (q_1, w_2)$, $h_3 = (q_2, w_1)$, $h_4 = (q_2, w_2)$, $h_5 = (q_3, w_1)$, $h_6 = (q_3, w_2)$.

Найдем отображение Rh_1 по входной букве z_1 . Состояние h_1 – это пара (q_1, w_1) , а входная буква z_1 – это пара (x_1, u_1) . Автомат A из состояния q_1 по входной букве x_1 согласно автоматной таблице перейдет в состояние q_2 , а автомат B из состояния w_1 по входной букве u_1 перейдет в состояние w_1 . Эта пара (q_2, w_1) согласно обозначениям есть состояние h_3 автомата K . На выходах автоматов A и B появятся при этом буквы y_1 и v_2

соответственно. Пара (y_1, v_2) – это буква s_2 выходного алфавита автомата K . Аналогично находим отображение Rh_1 по входной букве z_3 . Отображение состояния w_1 автомата B по входной букве u_2 не определено, поэтому также не определено отображение состояния h_1 автомата K по входным буквам z_2 и z_4 . Таким образом, получаем

$$Rh_1 = \{h_3(z_1, s_2), h_1(z_3, s_4)\}.$$

Далее проделываем то же самое для состояний h_2, h_3, h_4, h_5, h_6 . Соответствующие отображения этих состояний имеют вид

$$Rh_2 = \{h_3(z_1, s_1), h_4(z_2, s_1), h_1(z_3, s_3), h_2(z_4, s_3)\};$$

$$Rh_3 = \{h_5(z_1, s_4)\};$$

$$Rh_4 = \{h_5(z_1, s_3), h_6(z_2, s_3)\};$$

$$Rh_5 = \{h_5(z_3, s_2)\};$$

$$Rh_6 = \{h_5(z_3, s_1), h_6(z_4, s_1)\}.$$

По полученным отображениям можно составить автоматную таблицу:

K				
$h \backslash z$	z_1	z_2	z_3	z_4
h_1	h_3, s_2	-	h_1, s_4	-
h_2	h_3, s_1	h_4, s_1	h_1, s_3	h_2, s_3
h_3	h_5, s_4	-	-	-
h_4	h_5, s_3	h_6, s_3	-	-
h_5	-	-	h_5, s_2	-
h_6	-	-	h_5, s_1	h_6, s_1

и матрицу соединений \mathbf{R}_K :

\mathbf{R}_K

$h \backslash h$	h_1	h_2	h_3	h_4	h_5	h_6
h_1	z_3, s_4	0	z_1, s_2	0	0	0
h_2	z_3, s_4	z_4, s_3	z_1, s_1	z_2, s_1	0	0
h_3	0	0	0	0	z_1, s_4	0
h_4	0	0	0	0	z_1, s_3	z_2, s_3
h_5	0	0	0	0	z_3, s_2	0
h_6	0	0	0	0	z_3, s_1	z_4, s_1

Эту же матрицу соединений можно получить и из матриц соединений автоматов A и B . Для этого по автоматным таблицам составим матрицы соединений автоматов A и B – \mathbf{R}_A и \mathbf{R}_B :

 \mathbf{R}_A

	q_1	q_2	q_3
q_1	x_2, y_2	x_1, y_1	0
q_2	0	0	x_1, y_2
q_3	0	0	x_2, y_1

 \mathbf{R}_B

	w_1	w_2
w_1	u_1, v_2	0
w_2	u_2, v_1	u_1, v_1

Проведем прямое умножение этих матриц, т.е. каждый элемент одной матрицы умножается на каждый элемент другой матрицы (декартово произведение). В результате получим матрицу соединений автомата K (ввиду большого объема приведен только фрагмент матрицы):

 \mathbf{R}_K

	(q_1, w_1)	(q_1, w_2)	(q_2, w_1)	(q_2, w_2)
(q_1, w_1)	$(x_2, u_1), (y_2, v_2)$	0	$(x_1, u_1), (y_1, v_2)$	0
(q_1, w_2)	$(x_2, u_2), (y_2, v_1)$	$(x_2, u_1), (y_2, v_1)$	$(x_1, u_2), (y_1, v_1)$	$(x_1, u_1), (y_1, v_1)$
(q_2, w_1)	0	0	0	0
(q_2, w_2)	0	0	0	0

После переобозначений матрица принимает уже знакомый вид.

Автомат $K=A \times B$ соответствует параллельной одновременной работе автоматов A и B (рис. 4.9), причем Z и S определяются по формулам (4.4.13) и (4.4.15).

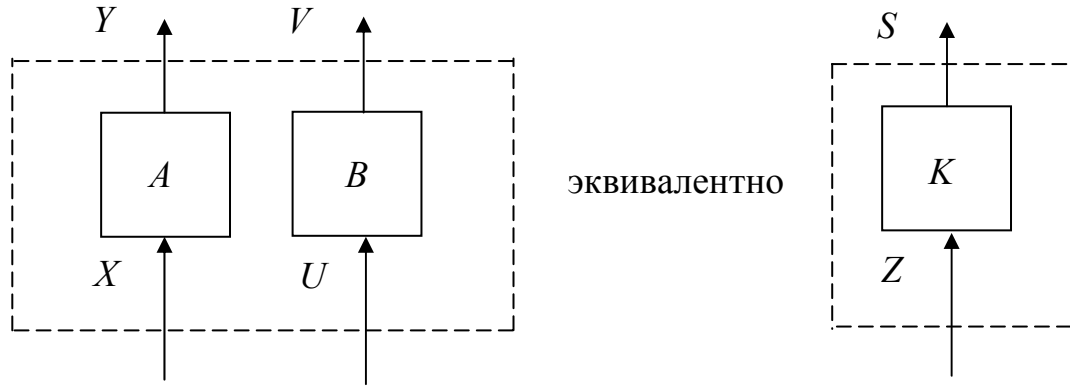


Рис. 4.9. Произведение двух автоматов с раздельными входами

Обозначим отображения, индуцируемые автоматами A и B через S_A и S_B соответственно, а отображение, индуцируемое автоматом $K=A \times B$, через S_K , и пусть $\mathbf{x} \in X^*$ и $\mathbf{u} \in U^*$ – слова в соответствующих алфавитах, имеющие равную длину:

$$\mathbf{x} = x_{i1} x_{i2} \dots x_{ik}, \quad \mathbf{u} = u_{j1} u_{j2} \dots u_{jk}$$

$$\text{и } S_A(\mathbf{x}) = y_{i1} y_{i2} \dots y_{ik}, \quad S_B(\mathbf{u}) = v_{j1} v_{j2} \dots v_{jk}.$$

Слово $\mathbf{z} \in Z^*$ является декартовым (прямым) произведением слов \mathbf{x} и \mathbf{u} и обозначается $\mathbf{z} = \mathbf{x} \times \mathbf{u}$, если каждая буква слова \mathbf{z} есть пара, образованная соответствующими буквами слов \mathbf{x} и \mathbf{u} . Поэтому $\mathbf{z} = (x_{i1}, u_{j1})(x_{i2}, u_{j2}) \dots (x_{ik}, u_{jk})$, и $S_K(\mathbf{z}) = (y_{i1}, v_{j1})(y_{i2}, v_{j2}) \dots (y_{ik}, v_{jk})$.

Если областью определения частичного отображения S_A является множество допустимых слов $\mathbf{x} \in X^*$, а областью определения частичного отображения S_B – множество допустимых слов $\mathbf{u} \in U^*$, то областью определения частичного отображения S_K будет множество таких слов $\mathbf{z} \in Z^*$, которые построены из допустимых слов \mathbf{x} и \mathbf{u} и имеют одинаковую длину. Таким образом, можно записать

$$S_K(\mathbf{z})=S_K(\mathbf{x}\times\mathbf{u})=S_A(\mathbf{x})\times S_B(\mathbf{u})=\mathbf{y}\times\mathbf{v}=\mathbf{s},$$

где $\mathbf{y}\in Y^*$, $\mathbf{v}\in V^*$, $\mathbf{s}\in S^*$.

Отображение S_K называется произведением отображений S_A и S_B :

$$S_K=S_A\times S_B.$$

Рассмотрим вторую операцию умножения, применяемую к автоматам с одним и тем же входным алфавитом. Возьмем два произвольных непустых автомата Мили: $A=(X, Q, Y, q_1\in Q, F(x\in X/y\in Y))$ и $B=(X, W, U, w_1\in W, P(x\in X/u\in U))$. Автомат $K=(X, V, S, v_1\in V, R(x\in X/s\in S))$ называется произведением автоматов A и B : $K=A\otimes B$, если:

$$V=Q\times W; \quad (4.4.17)$$

$$S=Y\times U; \quad (4.4.18)$$

$$Rv = \bigcup_{x\in X} (F_x q \times P_x w), \quad (4.4.19)$$

где $v\in V$, $q\in Q$, $w\in W$, $v=(q, w)$, а $F_x q$ и $P_x w$ – отображения состояний q и w соответственно по букве входного алфавита $x\in X$.

Вопрос о полной или частичной определенности произведения \otimes решается так же, как и в случае произведения \times .

Возьмем теперь матрицы соединений автоматов A и B и представим их объединением матриц автономных автоматов по входным буквам:

$$\mathbf{R}_A = \bigcup_{x\in X} \mathbf{R}_{Ax} \quad \text{и} \quad \mathbf{R}_B = \bigcup_{x\in X} \mathbf{R}_{Bx}.$$

Тогда матрица соединений R_K автомата $K=A\otimes B$ будет равна

$$\mathbf{R}_K = \bigcup_{x\in X} (\mathbf{R}_{Ax} \times \mathbf{R}_{Bx}),$$

т.е. объединению прямых произведений матриц автономных автоматов.

Пример 4.18. Найдем произведение двух автоматов с общим входом:

A

$q \backslash x$	x_1	x_2
q_1	q_1, y_2	q_2, y_1
q_2	q_1, y_1	-

 B

$w \backslash x$	x_1	x_2
w_1	-	w_1, u_2
w_2	w_1, u_1	w_2, u_1

Автомат $K = A \otimes B$ будет иметь алфавит состояний $V = \{v_1, v_2, v_3, v_4\}$ и выходной алфавит $S = \{s_1, s_2, s_3, s_4\}$,

где $v_1 = (q_1, w_1)$, $v_2 = (q_1, w_2)$, $v_3 = (q_2, w_1)$, $v_4 = (q_2, w_2)$, $s_1 = (y_1, u_1)$, $s_2 = (y_1, u_2)$, $s_3 = (y_2, u_1)$, $s_4 = (y_2, u_2)$.

Декартово произведение отображений состояний автоматов A и B по входной букве x_1 будет

	x_1
(q_1, w_1)	-
(q_1, w_2)	$(q_1, w_1), (y_2, u_1)$
(q_2, w_1)	-
(q_2, w_2)	$(q_1, w_1), (y_1, u_1)$

То же самое по входной букве x_2 будет

	x_2
(q_1, w_1)	$(q_2, w_1), (y_1, u_2)$
(q_1, w_2)	$(q_2, w_2), (y_1, u_1)$
(q_2, w_1)	-
(q_2, w_2)	-

Объединение этих таблиц с учетом введенных обозначений даст таблицу автомата K :

 K

	x_1	x_2
v_1	-	(v_3, s_2)
v_2	(v_1, s_3)	(v_4, s_1)
v_3	-	-
v_4	(v_1, s_1)	-

Автомат $K=A\otimes B$ соответствует параллельной одновременной работе автоматов A и B (рисунок 4.10).

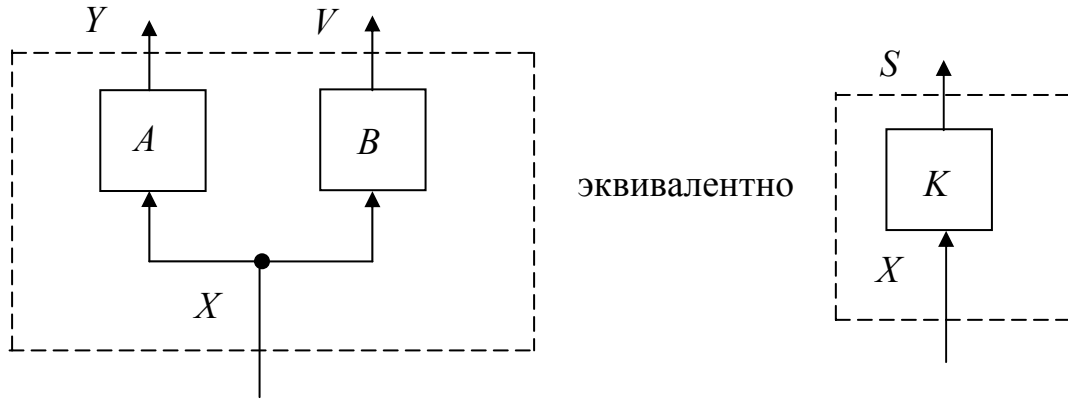


Рис. 4.10. Произведение двух автоматов с общим входом

Операцию умножения автоматов \times и \otimes можно обобщить и на случай n автоматов, а также использовать для нахождения произведений автоматов Мура.

Суммой двух произвольных автоматов $A=(X,Q,Y,q_1\in Q,F(x\in X/y\in Y))$ и $B=(U,W,V,w_1\in W,P(u\in U/v\in V))$ будет называться автомат $M=(Z,H,S,h\in H,R(z\in Z/s\in S))$ ($M=A+B$), если

$$Z=\{1\}\times X\cup\{2\}\times U; \quad (4.4.20)$$

$$H=Q\times W; \quad (4.4.21)$$

$$S=\{1\}\times Y\cup\{2\}\times V; \quad (4.4.22)$$

$$Rh=Fq\times\{w\}\cup\{q\}\times Pw, \quad (4.4.23)$$

где $q\in Q$, $w\in W$, $h\in H$, $h=(q,w)$. Начальным состоянием автомата M будет $h_1=(q_1,w_1)$.

Формулы (4.4.20) и (4.4.22) используются для того, чтобы различать, возможно, одинаковые буквы входных алфавитов X и U и выходных алфавитов Y и V . Если таких совпадающих букв нет, т.е.

$$X\cap U=\emptyset;$$

$$Y\cap V=\emptyset,$$

то вместо формул (4.4.20) и (4.4.22) используют выражения

$$Z=X \cup U; \quad (4.4.24)$$

$$S=Y \cup V. \quad (4.4.25)$$

Пример 4.19. Заданы два автомата A и B :

A		
	x_1	x_2
q_1	q_2, y_1	q_3, y_2
q_2	q_3, y_2	-
q_3	-	q_2, y_1

B		
	u_1	u_2
w_1	w_2, v_2	-
w_2	w_2, v_1	w_1, v_1

Найдем сумму $M=A+B$. Поскольку ни входные, ни выходные алфавиты автоматов A и B не пересекаются, для определения входного и выходного алфавитов автомата M пользуемся формулами (4.4.24) и (4.4.25). Поэтому $Z = \{x_1, x_2, u_1, u_2\}$ и $S = \{y_1, y_2, v_1, v_2\}$. Алфавит состояний, как и в случае операции произведения, будет $H = \{h_1, h_2, h_3, h_4, h_5, h_6\}$, где $h_1 = (q_1, w_1)$, $h_2 = (q_1, w_2)$, $h_3 = (q_2, w_1)$, $h_4 = (q_2, w_2)$, $h_5 = (q_3, w_1)$, $h_6 = (q_3, w_2)$.

Найдем отображение Rh_1 по входной букве x_1 , т.е. функцию перехода $\delta_M(h_1, x_1)$. Поскольку состояние h_1 есть пара (q_1, w_1) , а входная буква принадлежит алфавиту X автомата A , то в формуле (4.4.23) останется только первый член объединения, и $\delta_M(q_1, x_1) = (q_2, w_1) = h_3$. Функция выхода будет равна $\lambda_M(q_1, x_1) = y_1$.

Отображение Rh_1 по входной букве x_2 (функция перехода $\delta_M(q_1, x_2)$) будет равно $\delta_M(q_1, x_2) = (q_3, w_1) = h_5$, а функция выхода — $\lambda_M(q_1, x_2) = y_2$.

Отображение Rh_1 по входной букве u_1 (функция $\delta_M(h_1, u_1)$) будет представлено только вторым членом объединения в формуле (4.4.23), т.е. $\delta_M(q_1, u_1) = (q_1, w_2) = h_2$, а функция выхода будет равна $\lambda_M(q_1, u_1) = v_2$.

Функции перехода и выхода автомата B по входной букве u_2 не определены, поэтому не определено, конечно, и отображение Rh_1 по букве u_2 .

Окончательно будет

$$Rh_1 = \{h_3(x_1, y_1), h_5(x_2, y_2), h_2(u_1, v_2)\}.$$

Аналогичным образом определяем и отображения оставшихся состояний:

$$Rh_2 = \{h_4(x_1, y_1), h_6(x_2, y_2), h_2(u_1, v_1), h_1(u_2, v_1)\};$$

$$Rh_3 = \{h_5(x_1, y_2), h_4(u_1, v_2)\};$$

$$Rh_4 = \{h_6(x_1, y_2), h_4(u_1, v_1), h_3(u_2, v_1)\};$$

$$Rh_5 = \{h_3(x_2, y_1), h_6(u_1, v_1)\};$$

$$Rh_6 = \{h_4(x_2, y_1), h_6(u_1, v_1), h_5(u_2, v_1)\}.$$

По полученным отображениям составляем автоматную таблицу и матрицу соединений \mathbf{R}_M :

M

	x_1	x_2	u_1	u_2
h_1	h_3, y_1	h_5, y_2	h_2, v_2	-
h_2	h_4, y_1	h_6, y_2	h_2, v_1	h_1, v_1
h_3	h_5, y_2	-	h_4, v_2	-
h_4	h_6, y_2	-	h_4, v_1	h_3, v_1
h_5	-	h_3, y_1	h_6, v_1	-
h_6	-	h_4, y_1	h_6, v_1	h_5, v_1

\mathbf{R}_M

	h_1	h_2	h_3	h_4	h_5	h_6
h_1	0	u_1, v_2	x_1, y_1	0	x_2, y_2	0
h_2	u_2, v_1	u_1, v_1	0	x_1, y_1	0	x_2, y_2
h_3	0	0	0	u_1, v_2	x_1, y_2	0
h_4	0	0	u_2, v_1	u_1, v_1	0	x_1, y_2
h_5	0	0	x_2, y_1	0	0	u_1, v_1
h_6	0	0	0	x_2, y_1	u_2, v_1	u_1, v_1

Операцию суммирования и соответствующие формулы (4.4.20) – (4.4.25) легко обобщить на случай n автоматов и использовать для автоматов Мура.

Матрица соединений \mathbf{R}_M автомата $M=A+B$ определяется по формуле

$$\mathbf{R}_M = \mathbf{R}_A \times \mathbf{E}_B \cup \mathbf{E}_A \times \mathbf{R}_B,$$

где \mathbf{E}_A и \mathbf{E}_B – единичные матрицы порядков \mathbf{R}_A и \mathbf{R}_B соответственно.

Автомат $M=A+B$ соответствует параллельной неодновременной работе двух автоматов A и B (рис. 4.11).

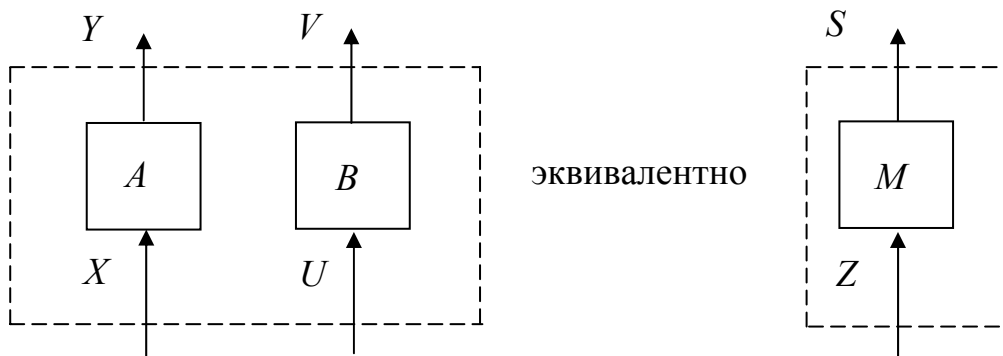


Рис. 4.11. Сумма двух автоматов

Любое входное слово в алфавите Z автомата M образуется чередованием букв x и u , принадлежащих алфавитам X и U . Аналогично и выходное слово автомата M – это последовательность чередующихся букв $y \in Y$ и $v \in V$. Если в очередной момент времени t на вход автомата M поступит бу-

ква $x \in X$, то в момент времени $t+1$ на входе обязательно появится буква $u \in U$. Выходная буква в момент времени t является буквой алфавита Y , а в момент времени $t+1$ – буквой из алфавита V . На уровне автоматов A и B это означает, что в момент времени t на входе автомата A имеется буква $x \in X$, а на входе автомата B в это же время – пустая буква e . В следующий $t+1$ -й момент времени на вход автомата B поступает буква $u \in U$, а на вход автомата A – пустая буква e .

Пусть $\mathbf{x} \in X^*$ и $\mathbf{u} \in U^*$ – слова, отличающиеся длиной не более чем на одну букву:

$$\mathbf{x} = x_{i_1} x_{i_2} \dots x_{i_k}, \quad \mathbf{u} = u_{j_1} u_{j_2} \dots u_{j_{k-1}}.$$

Слово $\mathbf{z} \in Z^*$ называется *сплетением* слов \mathbf{x} и \mathbf{u} , обозначается $\mathbf{z} = \mathbf{x} \langle \mathbf{u}$ и образуется чередованием букв \mathbf{x} и \mathbf{u} :

$$\mathbf{z} = \mathbf{x} \langle \mathbf{u} = x_{i_1} u_{j_1} x_{i_2} u_{j_2} \dots u_{j_{k-1}} x_{j_k}.$$

Любые две соседние буквы слова \mathbf{z} принадлежат разным входным алфавитам.

Если областью определения частичного отображения S_A служит множество допустимых слов $\mathbf{x} \in X^*$, а областью определения отображения S_B – множество допустимых слов $\mathbf{u} \in U^*$, то областью частичного отображения S_M будет множество слов $\mathbf{z} \in Z^*$, полученных сплетением пар допустимых слов \mathbf{x} и \mathbf{u} , отличающихся длиной не более чем на одну букву. Следовательно, можно записать

$$S_M(\mathbf{z}) = S_M(\mathbf{x} \langle \mathbf{u}) = S_A(\mathbf{x}) \langle S_B(\mathbf{u}) = \mathbf{y} \langle \mathbf{v} = \mathbf{s},$$

где $\mathbf{y} \in Y^*$, $\mathbf{v} \in V^*$, $\mathbf{s} \in S^*$. Отображение S_M называется *сплетением* отображений S_A и S_B и обозначается:

$$S_M = S_A \langle S_B.$$

Для произвольных автоматов A , B , и C выполняются следующие соотношения:

$$\begin{aligned}
(A \times B) \times C &= A \times (B \times C);, \\
A \times B &\sim B \times A; , \\
A \times E &\sim E \times A \sim A,
\end{aligned} \tag{4.4.26}$$

где роль единичного автомата E играет автомат $E=(X, Q, Y, q \in Q, F(x \in X/y \in Y))$, причем $X=\{x\}$, $Y=\{y\}$, $Q=\{q\}$, а $Fq=\{q(x/y)\}$.

Поэтому множество произвольных автоматов совместно с операцией умножения \times образует коммутативный моноид¹ с единичным элементом E . Обозначим это множество через B_{\times} .

Если брать множество автоматов с одним и тем же входным алфавитом, то очевидно, что для любых автоматов A , B , и C из этого множества выполняются соотношения, аналогичные (4.4.26), и, следовательно, это множество по операции \otimes образует коммутативный моноид с единичным элементом

$$E=(X, Q, Y, q \in Q, F(X \in X/y \in Y)),$$

где $X=\{x_1, x_2, \dots x_p\}$ – входной алфавит автоматов этого множества, $Q=\{q\}$, $Y=\{y\}$, $Fq=\{q(x_1 \cup x_2 \cup \dots \cup x_p/y)\}$. Это будет множество B_{\otimes} .

Аналогично множество произвольных автоматов по операции суммирования образуют коммутативную полугруппу (несущее множество B_{+}).

Следующая алгебраическая операция – суперпозиция двух автоматов. Возьмем два произвольных автомата из множества автоматов Мили, входящие и выходящие алфавиты которых могут пересекаться: $A=(X_1, Q, Y_1, q_1 \in Q, F(x \in X/y \in Y_1))$ и $B=(X_2, W, Y_2, w_1 \in W, P(x \in X_2/y \in Y_2))$, где пересечение $Y_1 \cap X_2 = Z$ не пусто в общем случае.

Отображение произвольного состояния $q \in Q$ автомата A можно представить в следующем виде:

¹ Моноидом в общей алгебре называют полугруппу с единицей E . Полугруппа, в свою очередь, – это множество элементов с заданной на этом множестве бинарной ассоциативной операцией, обычно называемой умножением. Таким образом, для любого элемента A моноида выполняется соотношение $E \cdot A = A \cdot E = A$.

$$Fq = F_{y_1} q \cup F_{y_2} q \cup \dots \cup F_{y_p} q = \bigcup_{y \in Y_1} F_y q,$$

где p – число букв входного алфавита Y_1 , а $F_y q$ – отображение состояния q , при котором на выходе появляется буква $y \in Y_1$.

Отображение произвольного состояния w автомата B представим в виде

$$Pw = P_{x_1} w \cup P_{x_2} w \cup \dots \cup P_{x_s} w = \bigcup_{x \in X_2} P_x w,$$

где s – число букв входного алфавита X_2 , а $P_x w$ – отображение состояния w по букве $x \in X_2$.

Автомат $N=(X_1, H, Y_2, h_1 \in H, S(x \in X_1 / y \in Y_2))$ будет являться суперпозицией автоматов A и B ($N=A*B$), если множество состояний

$$H=Q \times W, \quad (4.4.27)$$

а отображение S задано выражением

$$Sh = (F_{z_1} q \times P_{z_1} w) \cup (F_{z_2} q \times P_{z_2} w) \cup \dots \cup (F_{z_m} q \times P_{z_m} w) = \bigcup_{z \in Z_1} (F_z q \times P_z w),$$

где $h \in H, q \in Q, w \in W, h=(q, w), z \in Z=Y_1 \cap X_2$ (m – число букв алфавита Z).

Пример 4.20. Даны два автомата A и B :

A		
	x_1	x_2
q_1	q_2, y_1	q_3, y_2
q_2	q_3, y_2	■
q_3	■	q_2, y_1

B		
	y_1	y_2
w_1	w_2, v_2	■
w_2	w_2, v_1	w_1, v_1

Найдем суперпозицию $N = A * B$. Входной алфавит автомата N совпадает со входным алфавитом автомата A : $X = \{x_1, x_2\}$, а выходной алфавит – с выходным алфавитом автомата B : $V = \{v_1, v_2\}$. Алфавит состояний, как и во всех алгебраических операциях, – это $H = \{h_1, h_2, h_3, h_4, h_5, h_6\}$,

где $h_1 = (q_1, w_1)$, $h_2 = (q_1, w_2)$, $h_3 = (q_2, w_1)$, $h_4 = (q_2, w_2)$, $h_5 = (q_3, w_1)$, $h_6 = (q_3, w_2)$.

Возьмем состояние h_1 автомата N . Оно соответствует состоянию q_1 автомата A и состоянию w_1 автомата B . По входной букве x_1 автомат A перейдет из состояния q_1 в состояние q_2 с выходом y_1 , и по этой же букве y_1 автомат B перейдет из состояния w_1 в состояние w_2 с выходом v_2 . Поэтому пара (q_2, w_2) есть значение функции перехода автомата N : $\delta_N(h_1, x_1) = (q_2, w_2) = h_4$, а v_2 есть значение функции выхода автомата N : $\lambda_N(h_1, x_1) = v_2$. По входной букве x_2 автомат A переходит из состояния q_1 в состояние q_3 с выходом y_2 , но отображение состояния w_1 автомата B по этой букве (y_2) не определено, поэтому не определено и отображение состояния h_1 автомата N по входной букве x_2 . Таким образом, получаем

$$Sh_1 = \{h_4(x_1, v_2)\}.$$

Рассуждая аналогичным образом, получим отображения остальных состояний:

$$Sh_2 = \{h_4(x_1, v_1), h_5(x_2, v_1)\}, \quad Sh_3 = \emptyset, \quad Sh_4 = \{h_5(x_1, v_1)\},$$

$$Sh_5 = \{h_4(x_2, v_2)\}, \quad Sh_6 = \{h_4(x_2, v_1)\}.$$

Автоматная таблица, составленная по этим отображениям, приведена ниже:

	x_1	x_2
h_1	h_4, v_2	-
h_2	h_4, v_1	h_5, v_1
h_3	-	-
h_4	h_5, v_1	-
h_5	-	h_4, v_2
h_6	-	h_4, v_1

Суперпозицию автоматов можно представить и через разложение исходных автоматов на автономные. Представим автомат A объединением автономных автоматов по выходной букве:

$$A = \bigcup_{y \in Y_1} A_y,$$

а автомат B объединением автономных автоматов по входной букве:

$$B = \bigcup_{x \in X_2} B_x.$$

Тогда автомат $N=A*B$ будет задан выражением

$$N = (A_{z_1} \times B_{z_1}) \cup (A_{z_2} \times B_{z_2}) \cup \dots \cup (A_{z_n} \times B_{z_n}) = \bigcup_{z \in Z_1} (A_z \times B_z), \quad (4.4.29)$$

где $Z=Y_1 \cap X_2$.

Из последнего выражения следует, что операция суперпозиции автоматов соответствует композиции соответствующих отображений, если $Y_1=X_2$, т. е.

$$S_N(\mathbf{x})=S_B(\mathbf{y})=S_B(S_A(\mathbf{x})), \text{ или } S_N = S_A \circ S_B,$$

где $\mathbf{x} \in X_1$, $\mathbf{y} \in Y_1$.

Если же $Y_1 \neq X_2$ и $Y_1 \cap X_2 = Z$, то получим композицию сужений отображений S_A и S_B на множество Z . Операция суперпозиции автоматов ассоциативна, но, конечно же, некоммутативна, так же как и композиция отображений. Поэтому множество автоматов по операции суперпозиции образует некоммутативную полугруппу B_* .

Теперь запишем операцию суперпозиции в матричной форме.

Разложим матрицу соединений \mathbf{R}_A автомата A по автономным матрицам выходного алфавита:

$$\mathbf{R}_A = \bigcup_{y \in Y_1} \mathbf{R}_{A_y},$$

и из каждой автономной матрицы исключим ту букву, по которой выделена эта матрица.

Матрицу автомата B представим объединением матриц автономных автоматов входного алфавита:

$$\mathbf{R}_B = \bigcup_{x \in X_2} \mathbf{R}_{Bx},$$

также исключая из каждой автономной матрицы ту букву, по которой данная матрица выделена.

Тогда матрица соединений \mathbf{R}_N автомата $N=A*B$ при условии, что $Y_1 \cap X_2 = Z \neq \emptyset$, определяется формулой

$$\mathbf{R}_N = \bigcup_{z \in Z} (\mathbf{R}_{A_z} \times \mathbf{R}_{B_z}).$$

Введем понятие единичного и обратного автоматов на множестве произвольных автоматов. Под единичным автоматом E понимается такой автомат, который любое слово входного алфавита преобразует в такое же входное слово. Такой автомат индуцирует тождественное отображение на множестве слов X^* алфавита X . Ясно, что это автомат с единственным состоянием, матрица соединений которого имеет вид

$$\mathbf{R}_E = \|x_1 / x_1 \cup x_2 / x_2 \cup \dots \cup x_n / x_n\|.$$

Из определения единичного автомата E следует, что такой автомат является правой и левой единицей в полугруппе B_* :

$$A * E \sim E * A \sim A.$$

Обратным автоматом A^{-1} к автомату A , индуцирующему отображение $S_A(x)$ множество слов $x \in X^*$ на множество выходных слов $y \in Y^*$, называется такой автомат, который индуцирует обратное отображение $S_A^{-1}(y)$. Очевидно, что обратный автомат существует только тогда, когда отображение S_A является биективным (взаимнооднозначным) отображением. В этом случае в любой строке его матрицы соединений не может быть двух пар “вход-выход”, имеющих одинаковые выходные буквы. Поэтому для построения матрицы соединений обратного автомата достаточно в матрице

соединений автомата A в каждой паре “вход-выход” поменять местами элементы этой пары.

Таким образом, подмножество автоматов $D_* \subset B_*$, индуцирующих взаимнооднозначное отображение, по операции суперпозиции образует некоммутативную группу¹ D_* .

Операции суперпозиции двух автоматов соответствует последовательная их работа (рис. 4.12).

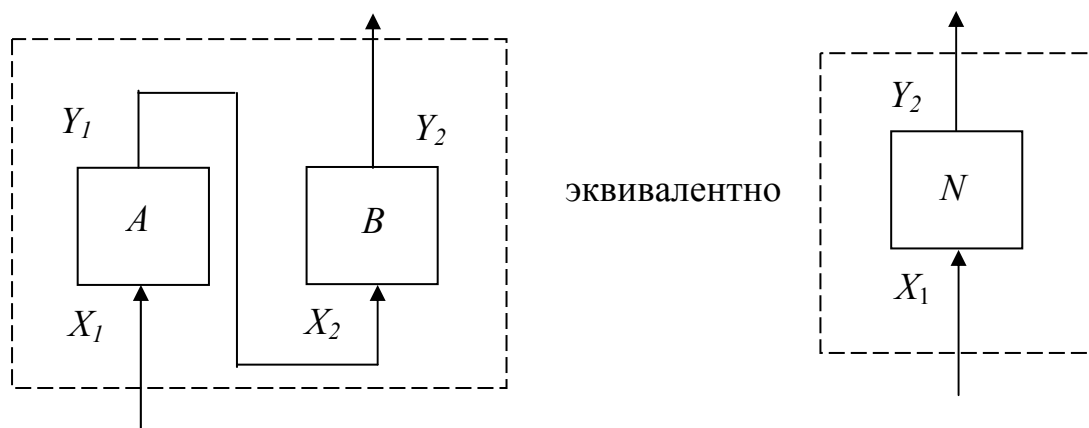


Рис. 4.12. Суперпозиция двух автоматов

Наиболее общий случай совместной работы автоматов задается операцией *композиции*, а различные типы их соединений и виды работы, которые соответствуют введенным ранее операциям, являются частными случаями композиции автоматов.

Зададим произвольные автоматы $A=(X,V,L, v_1 \in V, F(x \in X, t \in T/l \in L))$ и $B=(Y,W,T, w_1 \in W, P(y \in Y, l \in L/t \in T))$. Входной алфавит автомата A – это X , V – алфавит состояний, L – выходной алфавит, F – отображение множества состояний V в себя по буквам входного алфавита $x \in X$ и выходного алфави-

¹ Группой в общей алгебре называется полугруппа с единицей (моноид), для каждого элемента A которой существует обратный элемент A^{-1} , так, что $A^{-1} \cdot A = E$.

та $t \in T$ автомата B , при котором на выходе автомата A появляется выходная буква $l \in L$. Аналогично для автомата B .

Представим отображение F и P в виде объединения соответствующих выражений по буквам алфавитов T и L :

$$Fv = \bigcup_{t \in T} \bigcup_{l \in L} F_{t/l}v,$$

$$Pw = \bigcup_{l \in L} \bigcup_{t \in T} P_{l/t}w.$$

Автомат $C=(Z, Q, M, q_1 \in Q, K(z \in Z/m \in M))$ будет называться композицией автоматов A и B ($C = A \circ B$), если

$$Z = X \times Y; \quad (4.4.30)$$

$$Q = V \times W; \quad (4.4.31)$$

$$M = L \times T; \quad (4.4.32)$$

$$Kq = \bigcup_{\substack{l \in L \\ t \in T}} (F_{t/l}v \times P_{l/t}w), \quad (4.4.33)$$

где $q=(v,w)$, $q \in Q$, $v \in V$, $w \in W$, а $F_{t/l}v$ – отображение состояния v по букве $t \in T$, при котором на выходе появляется буква $l \in L$, $P_{l/t}w$ – отображение состояния w по букве $l \in L$, при котором на выходе появляется буква $t \in T$.

Композиции автоматов эквивалентна совместная работа автоматов, приведенная на рис. 4.13.

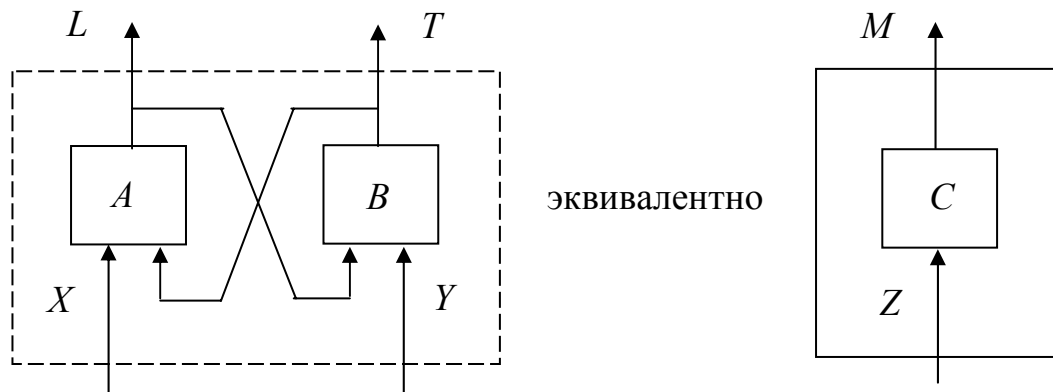


Рис. 4.13. Композиция двух автоматов

Автомат A можно разложить на автономные автоматы по входным буквам $t \in T$:

$$A = \bigcup_{t \in T} A_t,$$

а каждый автономный автомат A_t , свою очередь, – по буквам $l \in L$:

$$A_t = \bigcup_{l \in L} A_{t/l}.$$

Таким образом:

$$A = \bigcup_{\substack{t \in T \\ l \in L}} A_{t/l}. \quad (4.4.34)$$

Аналогично и автомат B :

$$B = \bigcup_{\substack{t \in T \\ l \in L}} B_{l/t}. \quad (4.4.35)$$

Из формул (4.4.30) – (4.4.35) следует, что автомат $C = A \circ B$ можно найти по формулам

$$C = \bigcup_{\substack{t \in T \\ l \in L}} A_{t/l} \times B_{l/t}. \quad (4.4.36)$$

Из последнего выражения легко получить операцию композиции в матричной форме. Возьмем матрицу соединений автомата A с элементами:

$$r_{\alpha\beta}(xt/l) = \begin{cases} x/l, & \text{если } v_\beta \in F_{v_\alpha} \text{ по буквам } x \in X, t \in T \text{ с выходом } l \in L, \\ 0, & \text{если } v_\beta \notin F_{v_\alpha}, \alpha, \beta = \{1, 2, \dots, k\}, \end{cases}$$

где k – число состояний автомата A , и матрицу соединений автомата B :

$$r_{\gamma\delta}(yl/t) = \begin{cases} yl/t, & \text{если } w_\delta \in P_{w_\gamma} \text{ по буквам } y \in Y, l \in L \text{ с выходом } t \in T, \\ 0, & \text{если } w_\delta \notin P_{w_\gamma}, \gamma, \delta = \{1, 2, \dots, n\}, \end{cases}$$

где n – число состояний автомата B .

Тогда матрица соединений автомата $C = A \circ B$ будет равна:

$$\mathbf{R}_C = \mathbf{R}_A \circ \mathbf{R}_B = \bigcup_{\substack{t \in T \\ l \in L}} \mathbf{R}_{A(t/l)} \times \mathbf{R}_{B(l/t)},$$

где $\mathbf{R}_{A(t/l)}$ – матрица соединений автономного подавтомата $A_{t/l}$ и буква $t \in T$, по которой выделен этот подавтомат, исключена из матрицы; $\mathbf{R}_{B(l/t)}$ – матрица соединений автономного подавтомата $B_{l/t}$, а буква $l \in L$ исключена.

В более общем случае автоматы A и B могут иметь и общий входной алфавит N : $A=(N, X, V, L, v_1 \in V, F(n \in N, x \in X, t \in T/l \in L))$, $B=(N, Y, W, T, w_1 \in W, P(n \in N, y \in Y, l \in L/t \in T))$. Тогда композиция A и B определяется выражением

$$C = A \circ B = \bigcup_{n \in N} (A_n \circ B_n),$$

где A_n и B_n – автономные автоматы по буквам входного алфавита $n \in N$.

Учитывая (4.4.36) получим

$$C = \bigcup_{n \in N} \left(\bigcup_{\substack{t \in T \\ l \in L}} (A_{n(t/l)} \times B_{n(l/t)}) \right),$$

то есть автомат C задан выражениями

$$Z = N \times X \times Y; \quad (4.4.37)$$

$$Q = V \times W; \quad (4.4.38)$$

$$M = L \times T; \quad (4.4.39)$$

$$Kq = \bigcup_{n \in N} \left(\bigcup_{\substack{t \in T \\ l \in L}} (F_{n(t/l)} v \times P_{n(l/t)} w) \right). \quad (4.4.40)$$

Используя соотношения (4.4.37) – (4.4.40), можно показать, что операция композиции ассоциативна и с точностью до изоморфизма коммутативна, и, следовательно, множество произвольных автоматов и заданная на этом множестве операция композиции образует коммутативную полу-группу B_\circ .

В частных случаях операция композиции соответствует рассмотренным ранее операциям умножения, суммирования, суперпозиции.

Например, пусть $A=(X, V, L, v_1 \in V, F(x \in X/l \in L))$ и $B=(Y, W, T, w_1 \in W, P(y \in Y/t \in T))$ – независимо работающие автоматы. Так как автоматы имеют

различные входные алфавиты ($X \cap Y = \emptyset$), пользуемся формулами (4.4.30) – (4.4.33). Отображение F состояния $v \in V$ автомата A не зависит от выхода автомата B , а отображение P состояния $w \in W$ автомата B не зависит от выхода автомата A , поэтому

$$\bigcup_{\substack{l \in L \\ t \in T}} F_{t/l} v = Fv, \quad \bigcup_{\substack{l \in L \\ t \in T}} P_{l/t} w = Pw.$$

Окончательно автомат $C=(Z, Q, M, q_1 \in Q, K(z \in Z/m \in M))$ определяется по формулам

$$Z=X \times Y, Q=V \times W, M=L \times T, Kq=Fv \times Pw,$$

которые совпадают с формулами (4.4.13) – (4.4.16) и, следовательно, задают операцию умножения \times .

Нетрудно для частных случаев перейти от композиции и к другим алгебраическим операциям: умножению \otimes автоматов с общим выходом, суперпозиции и суммированию.

4.4.3. Операции над вероятностными автоматами

Автоматы, которые до сих пор рассматривались, можно отнести к неслучайным или детерминированным автоматам в том смысле, что состояния таких автоматов в текущий момент времени однозначно определялось состоянием в предшествующий момент времени и буквами входного и выходного алфавитов в текущий момент времени. Наряду с такими автоматами естественно было бы рассматривать вероятностные автоматы, в которых переход из состояния в состояние носил бы стохастический или вероятностный характер. В реальных ситуациях это возможно из-за сбоев электронных схем при различного рода помехах.

Для простоты изложения рассмотрим вероятностные автоматы без выходов.

Вероятностный автомат считается заданным, если определена совокупность объектов:

$$A=(X, Q, q_1 \in Q, \varphi(q, x)),$$

где $X=\{x_1, x_2, \dots x_m\}$ – как обычно, входной алфавит, $Q=\{q_1, q_2, \dots q_n\}$ – алфавит состояний, $q_1 \in Q$ – начальное состояние автомата, $\varphi(q, x)$ – двуместная функция, задающая отображение множества $Q \times X$ в множество матриц $\mathbf{P}=\{\mathbf{P}_j\}$, $i=\{1, 2, \dots m\}$.

Эта функция $\varphi(q, x)$ называется таблицей переходных вероятностей. Для каждой пары (q, x) такой таблицы имеет место

$$\varphi(q, x)=\{p_1(q, x), p_2(q, x), \dots p_n(q, x)\}, \quad (4.4.41)$$

где $p_i(q, x)$ означает вероятность перехода в состояние q_i из состояния q по входной букве x и, следовательно, является неотрицательной величиной $p_i(q, x) \geq 0$ и удовлетворяет условию нормировки $\sum_{i=(1, \dots n)} p_i(q, x) = 1$.

Из записи (4.4.41) следует, что любая матрица $P_j \in P$ имеет вид $\mathbf{P}_j = \|p_{ik}(x)\|$ или, что то же самое, $\mathbf{P}_j = \|p_k(q_i, x)\|$, где $i, k \in \{1, 2, \dots n\}$. Все элементы p_{ik} каждой из матриц суть неотрицательные вещественные числа, не превосходящие единицы, и сумма элементов любой из строк равна единице. Предполагается также, что в автомате нет состояний, вероятности перехода в которые из всех других состояний равны нулю (т.е. нет столбцов, состоящих из одних нулей). Матрицы с такими свойствами называются стохастическими матрицами.

Если вероятностные автоматы рассматриваются в плане представления событий, то, как и для детерминированных автоматов, задается множество заключительных состояний $Q_z \subseteq Q$.

В вероятностном автомате можно вместо функции $\varphi(q, x)$ задать множество стохастических матриц \mathbf{P} . Тогда он будет записан в форме

$$A=(X, Q, q_1 \in Q, \mathbf{P}).$$

Обычный недетерминированный автомат можно рассматривать как частный случай вероятностного автомата, в котором для любого фиксированного $i \in \{1, 2, \dots, n\}$ только одна из вероятностей $p_{ik}(x)$ равна единице, а остальные равны нулю.

Нетрудно подсчитать вероятность перехода из некоторого состояния q_i в состояние q_k при поступлении на вход вероятностного автомата слова $\mathbf{x} \in X^*$. Действительно, пусть $\mathbf{x} = x_{j1}x_{j2}\dots x_{jl}$. Тогда вероятности перехода $p_{ik}(\mathbf{x})$ вычисляются умножением стохастических матриц $\mathbf{P}_{j1}\mathbf{P}_{j2}\dots\mathbf{P}_{jl}$:

$$\mathbf{P} = \mathbf{P}_{j1} \cdot \mathbf{P}_{j2} \cdot \dots \cdot \mathbf{P}_{jl} = \|p_{ik}(\mathbf{x})\|,$$

где $i, k \in \{1, 2, \dots, n\}$.

В последнем выражении применено обычное умножение (строка на столбец) матриц, которое возможно, поскольку матрицы \mathbf{P}_{ik} согласованы по форме (все они квадратные размерности $n \times n$).

Пример 4.21. Задан вероятностный автомат $A = (X, Q, q_1 \in Q, \mathbf{P})$, где

$$\mathbf{P}_{x_1} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{3}{4} & \frac{1}{4} \end{pmatrix}, \quad \mathbf{P}_{x_2} = \begin{pmatrix} 0 & 1 \\ \frac{1}{3} & \frac{2}{3} \end{pmatrix}.$$

Найдем вероятность перехода автомата из состояния q_1 в состояние q_2 при поступлении на вход автомата слова $\mathbf{x} = x_1x_1x_2$. Последовательно перемножая матрицы $\mathbf{P}_{x_1}, \mathbf{P}_{x_1}$ и \mathbf{P}_{x_2} , получаем

$$\mathbf{P}_{x_1} \cdot \mathbf{P}_{x_1} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{3}{4} & \frac{1}{4} \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{3}{4} & \frac{1}{4} \end{pmatrix} = \begin{pmatrix} \frac{5}{8} & \frac{3}{8} \\ \frac{9}{16} & \frac{7}{16} \end{pmatrix},$$

$$\mathbf{P}_x = \mathbf{P}_{x_1} \cdot \mathbf{P}_{x_1} \cdot \mathbf{P}_{x_2} = \begin{pmatrix} \frac{5}{8} & \frac{3}{8} \\ \frac{9}{16} & \frac{7}{16} \end{pmatrix} \times \begin{pmatrix} 0 & 1 \\ \frac{1}{3} & \frac{2}{3} \end{pmatrix} = \begin{pmatrix} \frac{1}{8} & \frac{7}{8} \\ \frac{7}{48} & \frac{41}{48} \end{pmatrix}.$$

Искомая вероятность – это элемент первой строки и второго столбца последней матрицы, т.е. $\frac{7}{8}$.

Если $Q_z = \{q_\alpha\}$, $\alpha \in I' = \{1, 2, \dots, k\}$ – множество заключительных состояний, то вероятность:

$$p(\mathbf{x}) = \sum_{\alpha \in I'} p_{1\alpha}(\mathbf{x})$$

есть вероятность того, что автомат из начального состояния q_1 перейдет в одно из заключительных состояний $q \in Q_z$ при подаче на вход автомата слова \mathbf{x} .

Вероятностные автоматы можно задавать графами переходов, как и детерминированные автоматы, только нужно около каждого ребра, связывающего вершины q_i с q_j , ставить кроме буквы входного алфавита x еще и соответствующую вероятность перехода $p_{ij}(x)$. Понятно, что аналитический способ создания автоматов (4.4.41) и геометрическая интерпретация в виде графа неудобны и громоздки даже при небольшом числе букв входного алфавита, поэтому вероятностный автомат задают обычно системой стохастических матриц.

Используя такой способ задания вероятностных автоматов, можно ввести теоретико-множественные операции над ними по аналогии с операциями над детерминированными автоматами, но ограничения на стохастические матрицы, которые при этом нужно накладывать, делают класс автоматов, к которым применимы эти операции, довольно узким. Поэтому переходим сразу к алгебраическим операциям.

Зададим два вероятностных автомата $A = (X, Q, q_1 \in Q, \mathbf{P})$ и $B = (Y, V, v_1 \in V, \mathbf{S})$. Вероятностный автомат $C = (Z, W, w_1 \in W, \mathbf{R})$ будет являться произведением автоматов A и B ($C = A \times B$), если:

$$Z = X \times Y; \quad (4.4.42)$$

$$W = Q \times V; \quad (4.4.43)$$

$$\mathbf{R} = \mathbf{P} \times \mathbf{S}, \quad (4.4.44)$$

где $w_1 = (q_1, v_1)$. Формулы (4.4.42), (4.4.43) – это обычные декартовы произведения, а (4.4.44) – это декартово произведение, образуемое по правилу прямого произведения стохастических матриц из \mathbf{P} и \mathbf{S} .

Пример 4.22. Два вероятностных автомата $A = (X, Q, q_1 \in Q, \mathbf{P})$ и $B = (Y, V, v_1 \in V, \mathbf{S})$ заданы своими стохастическими матрицами:

$$\mathbf{P}_{x_1} = \begin{pmatrix} \frac{2}{3} & \frac{1}{3} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix}, \quad \mathbf{P}_{x_2} = \begin{pmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix},$$

$$\mathbf{S}_{y_1} = \begin{pmatrix} 1 & 0 \\ \frac{1}{5} & \frac{4}{5} \end{pmatrix}, \quad \mathbf{S}_{y_2} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix}.$$

Найдем автомат $C = A \times B$. Входной алфавит Z и алфавит состояний W автомата C , согласно формулам (4.4.42), (4.4.43), будут равны

$$Z = \{z_1, z_2, z_3, z_4\}, \quad W = \{w_1, w_2, w_3, w_4\},$$

где $z_1 = (x_1, y_1)$, $z_2 = (x_1, y_2)$, $z_3 = (x_2, y_1)$, $z_4 = (x_2, y_2)$, $w_1 = (q_1, v_1)$, $w_2 = (q_1, v_2)$, $w_3 = (q_2, v_1)$, $w_4 = (q_2, v_2)$.

Стохастические матрицы автомата C найдем по формуле (4.4.44) прямым произведением (элемент на элемент) соответствующих матриц автоматов A и B .

$$\mathbf{R}_{z_1} = \mathbf{P}_{x_1} \times \mathbf{S}_{y_1} = \begin{pmatrix} \frac{2}{3} & \frac{1}{3} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ \frac{1}{5} & \frac{4}{5} \end{pmatrix} = \begin{pmatrix} \frac{2}{3} & 0 & \frac{1}{3} & 0 \\ \frac{2}{15} & \frac{8}{15} & \frac{1}{15} & \frac{4}{15} \\ \frac{1}{4} & 0 & \frac{3}{4} & 0 \\ \frac{1}{20} & \frac{1}{5} & \frac{3}{20} & \frac{3}{5} \end{pmatrix},$$

$$\mathbf{R}_{z_2} = \mathbf{P}_{x_1} \times \mathbf{S}_{y_2} = \begin{pmatrix} \frac{2}{3} & \frac{1}{3} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix} \times \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{2} & \frac{1}{12} & \frac{1}{4} \\ \frac{1}{8} & \frac{1}{8} & \frac{3}{8} & \frac{3}{8} \\ \frac{1}{16} & \frac{3}{16} & \frac{3}{16} & \frac{9}{16} \end{pmatrix},$$

$$\mathbf{R}_{z_3} = \mathbf{P}_{x_2} \times \mathbf{S}_{y_1} = \begin{pmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ \frac{1}{5} & \frac{4}{5} \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & 0 & \frac{2}{3} & 0 \\ \frac{1}{15} & \frac{4}{15} & \frac{2}{15} & \frac{8}{15} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{10} & \frac{2}{5} & \frac{1}{10} & \frac{2}{5} \end{pmatrix},$$

$$\mathbf{R}_{z_4} = \mathbf{P}_{x_2} \times \mathbf{S}_{y_2} = \begin{pmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \times \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix} = \begin{pmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{12} & \frac{1}{4} & \frac{1}{6} & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{8} & \frac{3}{8} & \frac{1}{8} & \frac{3}{8} \end{pmatrix}.$$

Вероятностный автомат $D = (Z, W, w_1 \in W, \mathbf{R})$ является суммой автоматов A и B ($D=A+B$), если:

$$Z = (\{1\} \times X) \cup (\{2\} \times Y); \quad (4.4.45)$$

$$W = Q \times Y; \quad (4.4.46)$$

$$\mathbf{R} = (\mathbf{P} \times \mathbf{E}_B) \cup (\mathbf{E}_A \times \mathbf{S}), \quad (4.4.47)$$

где $w_1 = (q_1, v_1)$, а \mathbf{E}_A и \mathbf{E}_B – единичные матрицы, имеющие порядок матриц из P и S соответственно, причем произведения в круглых скобках выражения (4.4.47) являются прямыми произведениями. Попутно вспомним, что порядки стохастических матриц \mathbf{P} и \mathbf{S} (они, конечно, квадратные) равны соответствующим мощностям множеств Q и V .

Если входные алфавиты вероятностных автоматов A и B не пересекаются $X \cap Y = \emptyset$, то формулу (4.4.45) можно заменить на:

$$Z = X \cup Y. \quad (4.4.48)$$

Пример 4.23. Найдем сумму автоматов A и B из примера 4.22. Поскольку входные алфавиты автоматов A и B не пересекаются, то для определения входного алфавита автомата $D=A+B$ пользуемся формулой (4.4.48): $Z = \{x_1, x_2, y_1, y_2\}$. Далее по формуле (4.4.47) находим стохастические матрицы автомата D .

$$\mathbf{R}_{x_1} = \mathbf{P}_{x_1} \times \mathbf{E}_B = \begin{pmatrix} \frac{2}{3} & \frac{1}{3} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{2}{3} & 0 & \frac{1}{3} & 0 \\ 0 & \frac{2}{3} & 0 & \frac{1}{3} \\ \frac{1}{4} & 0 & \frac{3}{4} & 0 \\ 0 & \frac{1}{4} & 0 & \frac{3}{4} \end{pmatrix},$$

$$\mathbf{R}_{x_2} = \mathbf{P}_{x_2} \times \mathbf{E}_B = \begin{pmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & 0 & \frac{2}{3} & 0 \\ 0 & \frac{1}{3} & 0 & \frac{2}{3} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix},$$

$$\mathbf{R}_{y_1} = \mathbf{E}_A \times \mathbf{S}_{y_1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ \frac{1}{5} & \frac{4}{5} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{5} & \frac{4}{5} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{5} & \frac{4}{5} \end{pmatrix},$$

$$\mathbf{R}_{y_2} = \mathbf{E}_A \times \mathbf{S}_{y_2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{3}{4} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{4} & \frac{3}{4} \end{pmatrix}.$$

Операции умножения и суммирования легко обобщить на случай n автоматов.

Для задания операции суперпозиции будем полагать, что алфавит состояний первого автомата совпадает с входным алфавитом второго автомата, т.е. $A = (X, Q, q_1 \in Q, \mathbf{P})$ и $B = (Q, V, v_1 \in V, \mathbf{S})$, поскольку рассматриваются автоматы без выходов. Вероятностный автомат $C = (X, W, w_1 \in W, \mathbf{R})$ будет являться суперпозицией автоматов A и B ($C = A * B$), если

$$W = Q \times V; \quad (4.4.49)$$

$$\begin{aligned}
\mathbf{R} &= \bigcup_k (\mathbf{P}_{x_k}^{(q_1)} \times \mathbf{S}_{q_1}) \cup (\mathbf{P}_{x_k}^{(q_2)} \times \mathbf{S}_{q_2}) \cup \dots \cup (\mathbf{P}_{x_k}^{(q_n)} \times \mathbf{S}_{q_n}) = \\
&= \bigcup_k \bigcup_i \mathbf{P}_{x_k}^{(q_i)} \times \mathbf{S}_{q_i},
\end{aligned} \tag{4.4.50}$$

где $i \in \{1, 2, \dots, n\}$, $w_1 = (q_1, v_1)$, а $\mathbf{P}_{x_k}^{(q_i)}$ – матрица, полученная из стохастической матрицы \mathbf{P}_{x_k} заменой всех столбцов, отличных от q_i , нулевыми столбцами. По-прежнему в выражении (4.4.50) подразумевается прямое произведение соответствующих матриц.

Пример 4.24. Пусть даны автоматы $A = (X, Q, q_1 \in Q, \mathbf{P})$ и $B = (Q, V, v_1 \in V, \mathbf{S})$.

$$\begin{aligned}
\mathbf{P}_{x_1} &= \begin{pmatrix} \frac{2}{3} & \frac{1}{3} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix}, \quad \mathbf{P}_{x_2} = \begin{pmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}; \\
\mathbf{S}_{q_1} &= \begin{pmatrix} 1 & 0 \\ \frac{1}{5} & \frac{4}{5} \end{pmatrix}, \quad \mathbf{S}_{q_2} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix}.
\end{aligned}$$

Найдем автомат $C = (X, W, w_1 \in W, \mathbf{R})$, равный суперпозиции автоматов A и B ($C = A * B$). Согласно формуле (4.4.50), находим

$$\mathbf{R}_{x_1} = \mathbf{P}_{x_1}^{(q_1)} \times \mathbf{S}_{q_1} \cup \mathbf{P}_{x_1}^{(q_2)} \times \mathbf{S}_{q_2} = \begin{pmatrix} \frac{2}{3} & 0 \\ \frac{1}{4} & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ \frac{1}{5} & \frac{4}{5} \end{pmatrix} \cup \begin{pmatrix} 0 & \frac{1}{3} \\ 0 & \frac{3}{4} \end{pmatrix} \times \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix} =$$

$$\begin{aligned}
&= \begin{pmatrix} \frac{2}{3} & 0 & 0 & 0 \\ \frac{2}{15} & \frac{8}{15} & 0 & 0 \\ \frac{1}{4} & 0 & 0 & 0 \\ \frac{1}{20} & \frac{1}{5} & 0 & 0 \end{pmatrix} \cup \begin{pmatrix} 0 & 0 & \frac{1}{6} & \frac{1}{6} \\ 0 & 0 & \frac{1}{12} & \frac{1}{4} \\ 0 & 0 & \frac{3}{8} & \frac{3}{8} \\ 0 & 0 & \frac{3}{16} & \frac{9}{16} \end{pmatrix} = \begin{pmatrix} \frac{2}{3} & 0 & \frac{1}{6} & \frac{1}{6} \\ \frac{2}{15} & \frac{8}{15} & \frac{1}{12} & \frac{1}{4} \\ \frac{1}{4} & 0 & \frac{3}{8} & \frac{3}{8} \\ \frac{1}{20} & \frac{1}{5} & \frac{3}{16} & \frac{9}{16} \end{pmatrix}. \\
\\
\mathbf{R}_{x_2} &= \mathbf{P}_{x_2}^{(q_1)} \times \mathbf{S}_{q_1} \cup \mathbf{P}_{x_2}^{(q_2)} \times \mathbf{S}_{q_2} = \begin{pmatrix} \frac{1}{3} & 0 \\ \frac{1}{2} & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ \frac{1}{5} & \frac{4}{5} \end{pmatrix} \cup \begin{pmatrix} 0 & \frac{2}{3} \\ 0 & \frac{1}{2} \end{pmatrix} \times \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix} = \\
&= \begin{pmatrix} \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{15} & \frac{4}{15} & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{10} & \frac{2}{5} & 0 & 0 \end{pmatrix} \cup \begin{pmatrix} 0 & 0 & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & \frac{1}{6} & \frac{1}{2} \\ 0 & 0 & \frac{1}{4} & \frac{1}{4} \\ 0 & 0 & \frac{1}{8} & \frac{3}{8} \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{15} & \frac{4}{15} & \frac{1}{6} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{10} & \frac{2}{5} & \frac{1}{8} & \frac{3}{8} \end{pmatrix}.
\end{aligned}$$

С содержательной точки зрения операции над вероятностными автоматами означают то же самое, что и над детерминированными автоматами.

4.5. Структурное исследование автоматов

Переходим теперь к внутреннему устройству автоматов и к задачам, связанным с их внутренним устройством, то есть к структурному уровню. Здесь, как и на абстрактном уровне, главными задачами исследования являются анализ и синтез автоматов.

4.5.1. Комбинационные логические автоматы

Для дальнейшего изложения нужно дать несколько определений.

Автомат называется *комбинационным*, если для любого входного символа $x \in X$ и любых состояний $q_i, q_j \in Q$ выполняется равенство

$$\lambda(q_i, x) = \lambda(q_j, x),$$

то есть выход автомата не зависит от его состояния и определяется только его входом. В таком автомате все состояния эквиваленты и минимальный комбинационный автомат имеет *только одно состояние*. Функция переходов в нем вырождена, а поведение такого автомата задается функцией выходов с одним аргументом $\lambda(x_i) = y_i$.

Если входной алфавит автомата состоит из 2^m двоичных векторов длины m , а выходной – из 2^n двоичных векторов длины n , то такой автомат называется *логическим*. Понятно, что автомат с произвольными алфавитами можно свести к логическому автомату соответствующим *кодированием* его алфавитов. Таким образом, *комбинационный логический* автомат – это автомат, функция выхода которого – это система n логических функций от m переменных:

$$\begin{aligned} y_1 &= \lambda_1(x_1, x_2, \dots, x_m); \\ y_2 &= \lambda_2(x_1, x_2, \dots, x_m); \\ &\dots \\ y_n &= \lambda_n(x_1, x_2, \dots, x_m), \end{aligned} \tag{4.5.1}$$

где x_i – логическая переменная (i -я компонента вектора x длины m), а y_j – также логическая переменная (j -я компонента вектора y длины n).

Эту же систему уравнений (4.5.1) можно записать и в компактной форме $y = \Phi(x)$, где Φ – упорядоченная совокупность функций λ_i .

Логический комбинационный автомат можно представить рис. 4.14.

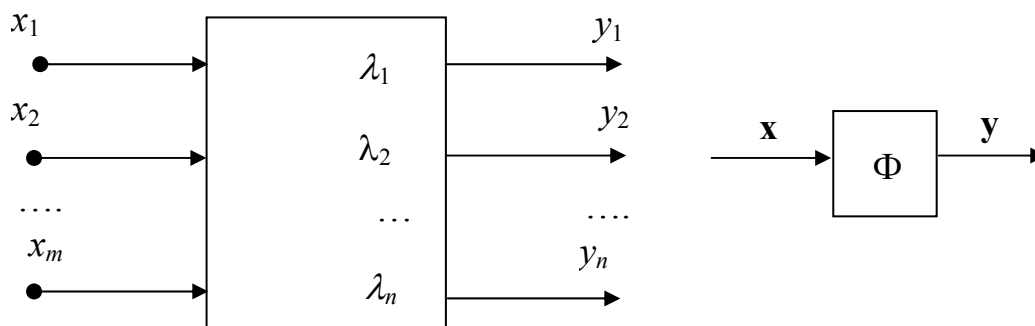


Рис. 4.14. Функциональная схема комбинационного автомата

Удобно рассматривать x_1, x_2, \dots, x_m как входные, а y_1, y_2, \dots, y_n — как выходные полюсы автомата. Считая, что каждый полюс может находиться в состоянии 0 или 1, приходим к выводу, что в комбинационном логическом автомате каждой комбинации состояний входных полюсов вполне однозначно соответствует комбинация состояний выходных полюсов. Отсюда и название — *комбинационный*. Система уравнений (4.5.1) и схема на рис. 4.14 — это *функциональная* модель автомата.

4.5.2. Постановка задач синтеза и анализа на структурном уровне

Структурная схема автомата, т. е. его структурная модель, показывает, как он устроен, из каких элементов состоит и как эти элементы соединены (связаны) между собой.

Структурная модель дискретного автомата отражает схему реальной дискретной системы, и элементы автомата ставятся в соответствие некоторым реальным конструктивным элементам.

Основное содержание теории автоматов на структурном уровне — это исследование соотношения между функциональной моделью и структурной моделью. И по-прежнему здесь возникают две задачи: задача анализа и задача синтеза. Задача анализа — это получение функциональной

модели по заданной структурной. Синтез – обратная задача: нахождение структурной модели по заданной функциональной. Вторая задача значительно сложнее, так как ее решение не единственно и среди многих возможных решений требуется выбрать оптимальное (наилучшее) в каком-то смысле. Поскольку задача синтеза более трудная, то и большая часть сил и времени на структурном уровне тратится на решение именно этой задачи.

Исходная для синтеза информация задается следующим образом. Во-первых, описывается функциональная модель. Во-вторых, указывается из каких элементов нужно автомат синтезировать, то есть задается *элементный базис*. В-третьих, определяется *синтаксис структур*, то есть формулируются правила взаимных соединений элементов, выделяющие из всевозможных структур класс допустимых (или правильных). Синтаксис играет компенсирующую роль: заменяя реальные элементы абстрактными, мы допускаем некоторую идеализацию, которая, тем не менее, оказывается допустимой, пока синтезированные из данных элементов структуры являются правильными, т. е. удовлетворяющими синтаксису. Однако как только мы переходим к рассмотрению неправильных структур, может появиться нежелательный эффект идеализации, то есть поведение реального устройства может существенно отклониться от поведения его абстрактного двойника (модели).

Будем считать, что элементный базис в совокупности с правилами соединения элементов образуют базис синтеза или просто базис.

4.5.3. Элементный базис

В элементный базис могут входить самые разнообразные элементы, которые сами являются простейшими автоматами. Выбор их диктуется как уровнем развития технологии производства реальных элементов, так и требованиями, предъявляемыми к базису со стороны методов синтеза. Ос-

новные требования, которым должен удовлетворять элементный базис – это *полнота и эффективность*.

Базис называется полным относительно некоторого класса автоматов, если в нем может быть синтезирован любой автомат этого класса.

Требование эффективности достаточно расплывчатое, и его можно сформулировать примерно так: более эффективным будет базис, синтезируемые в котором структуры будут в каком-то смысле лучшими (проще, дешевле, надежнее и т.д.). При определении эффективности базиса нужно учитывать как свойства реальных элементов, так и применяемые методы синтеза: для одних базисов эти методы могут быть развиты сильнее, для других – слабее.

Какие же элементы могут входить в базис? Это, во-первых, логические элементы и, во-вторых, – элементы памяти.

Логическими элементами называются элементарные комбинационные логические автоматы, функциональные свойства которых представляются достаточно простыми логическими функциями: дизъюнкцией, конъюнкцией, отрицанием, функцией Шеффера, импликацией, стрелкой Пирса и т.д.

Как правило, ограничиваются элементами И (конъюнкция), ИЛИ (дизъюнкция), НЕ (отрицание), И-НЕ (штрих Шеффера), ИЛИ-НЕ (стрелка Пирса) и многовходовыми аналогами соответствующих элементов.

Элементами памяти служат некоторые элементарные логические автоматы. Наиболее простые и распространенные из них – это *элемент задержки* и *триггер*.

Элемент задержки можно рассматривать как элементарный синхронный автомат, функции которого сводятся к задержке на один такт значения одной логической переменной. То есть значение выхода в момент времени t равно значению входа в момент времени $t-1$. Схематичное изображение и автоматная таблица элемента задержки приведены на рис. 4.15.

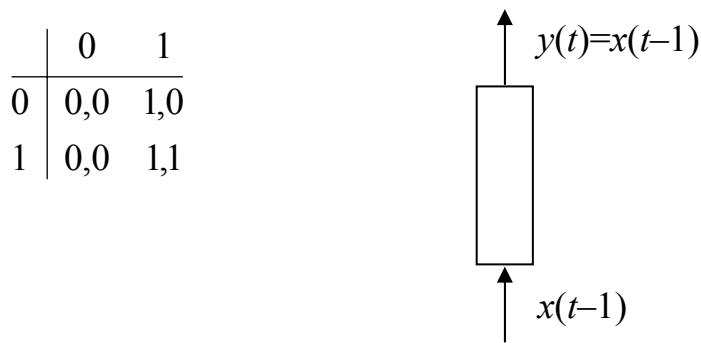


Рис. 4.15. Элемент задержки

Триггер можно представить себе как асинхронный автомат с двумя внутренними состояниями, которые могут фиксироваться и в каждое из которых при определенных условиях автомат можно перевести. Из различных вариантов автоматов, удовлетворяющих этим условиям, остановимся на автомате, графическое изображение и автоматная таблица которого приведены на рис. 4.16.

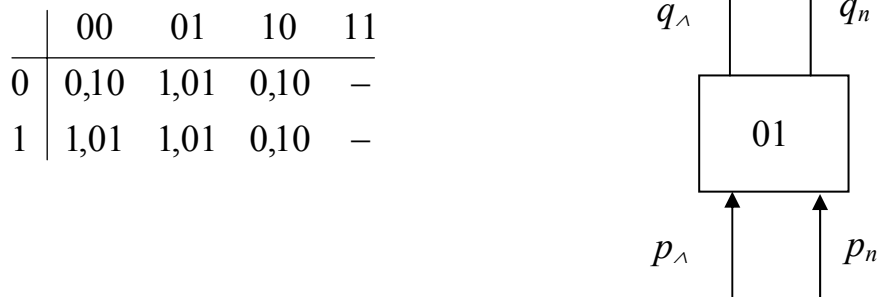


Рис. 4.16. Триггер

Входом и выходом такого автомата являются двоичные векторы длины 2. Первые компоненты этих векторов проиндексированы на рисунке 4.16 буквой \wedge (левые полюса), а вторые – буквой n (правые полюса). В связи с тем, что поведение триггера при комбинации 11 на входе не определено, использовать эту комбинацию на входе не рекомендуется.

4.5.4. Автоматные сети

Возьмем некоторую совокупность автоматных элементов (безразлично, разных или одинаковых). Выделим из множества входных полюсов P всех элементов некоторое подмножество $X \subset P$, а из множества Q всех выходных полюсов некоторое подмножество Y . Отобразим разность $P \setminus X$ в Q и будем считать, что это отображение задает множество связей между элементами множества P с одной стороны и множества Q – с другой. Полученную таким образом структуру будем называть *сетью*, элементы множества X – ее входными полюсами, а элементы множества Y – ее выходными полюсами. При небольшом числе элементов в сети можно пользоваться графическим представлением, в иных случаях удобнее задавать сеть в форме некоторого списка, содержащего перечень элементов и связей между ними.

Очевидно, что структуру любого автомата можно представить некоторой сетью. Обратное, вообще говоря, неверно. Для подтверждения этого факта достаточно рассмотреть классический пример сети, изображенной на рис. 4.17, а.

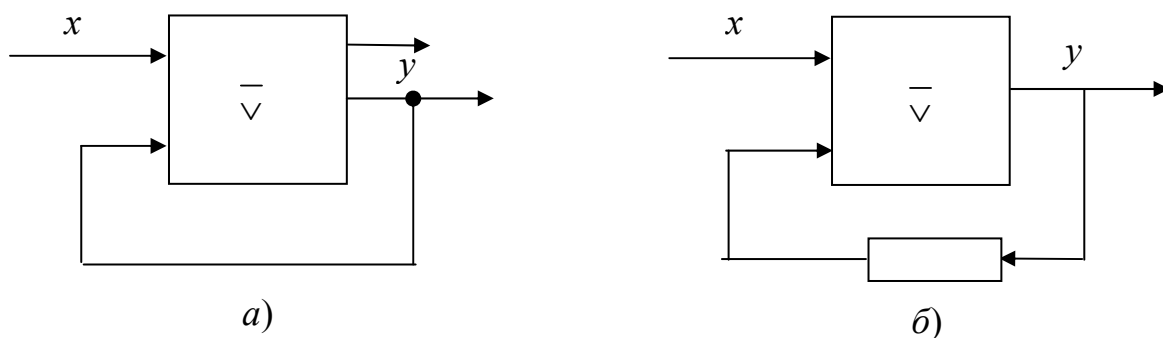


Рис. 4.17. Пример сети

Для $x=0$ при определении значения y возникает противоречие типа “если $y=0$, то $y=1$, если $y=1$, то $y=0$ ”. Это противоречие можно разрешить,

если добавить в обратную связь, например, блок задержки (рис. 4.17, б). В этом случае сеть можно рассматривать как синхронный автомат, в котором при $x=0$ переменная y принимает последовательно значения 1, 0, 1, 0,

Это пример показывает, что сети из логических элементов, содержащие контур обратной связи без задержек, могут не иметь конкретной автоматной интерпретации. В то же время обратные связи с элементами памяти являются мощным средством построения автоматов. В связи с этим будем рассматривать только *правильные* сети, то есть такие, которые можно рассматривать как структуры автоматов.

Правильная синхронная сеть – это сеть из логических элементов и элементов задержки (назовем и те и другие для краткости блоками), если: 1) к каждому входному полюсу блока присоединен не более, чем один выходной полюс (однако допускается присоединение выходного полюса блока к нескольким входным полюсам, то есть разветвление выходов); 2) в каждом контуре обратной связи, то есть в каждом цикле, есть хотя бы один элемент задержки. Входными полюсами правильной синхронной сети будут полюса, не присоединенные ни к каким выходным полюсам блоков, а выходными полюсами – только те, которые не подсоединены ни к каким входным полюсам.

Оказывается, что любой автомат можно представить правильной синхронной сетью. Об этом говорит следующая теорема.

Теорема 4.5.1. Любой конечный автомат при любом двоичном кодировании его алфавитов X , Q , Y может быть реализован правильной синхронной сетью из логических комбинационных автоматов и двоичных задержек, причем число задержек не может быть меньше $\log_2 |Q|$.

Доказательство. Действительно, автомат с произвольными функциями $q(t+1)=\delta(q(t), x(t))$, $y(t)=\lambda(q(t), x(t))$ может быть представлен сетью на рис. 4.18.

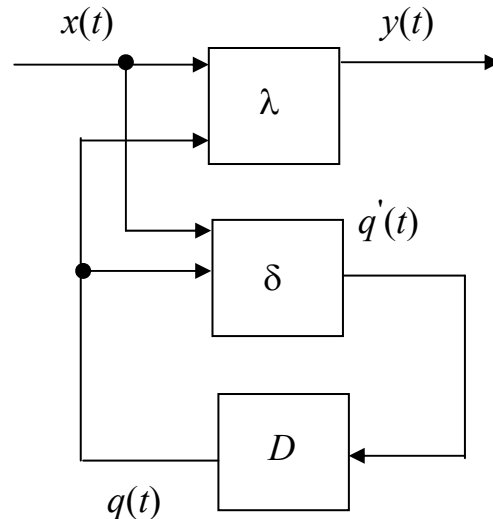


Рис. 4.18. Автомат, представленный правильной синхронной сетью

Здесь блок λ – комбинационный автомат с входным алфавитом $X \times Q$ и выходным алфавитом Y , вычисляющий функцию $y(t) = \lambda(q(t), x(t))$, блок δ – комбинационный автомат с тем же входным алфавитом и выходным алфавитом Q . Блок D – это блок, задерживающий поступающие на его вход сигналы на один такт. Его можно представить как автомат Мура, входной и выходной алфавиты которого совпадают и равны Q , алфавит состояний $R = \{r_1, r_2, \dots, r_n\}$ имеет ту же мощность, что и Q , $\lambda_D(r_i) = q_i$, $\delta_D(r_i, q_j) = r_j$, $i, j \in \{1, 2, \dots, n\}$. Сигнал $q'(t)$ на выходе блока δ – это вычисленное значение $\delta(q(t), x(t))$, которое, будучи задержанным блоком D на один такт, появляется на входах блоков δ и λ в момент времени $t+1$, то есть $q'(t) = \delta(q(t), x(t))$, которое, будучи задержанным блоком D на один такт, появляется на входах блоков δ и λ в момент времени $t+1$, то есть $q'(t) = \delta(q(t), x(t)) = q(t+1)$.

Осталось превратить сеть, изображенную на рис. 4.18, в правильную. Для этого требуется алфавиты X , Q , Y закодировать двоичными наборами, число входов и выходов блоков λ , δ и D согласовать с длинами соответствующих наборов, а блок D реализовать параллельным соединением двоичных задержек. Число этих задержек k равно длине двоичного вектора

q , а наименьшая длина этого вектора при двоичном кодировании n состояний составляет $\log_2 |Q|$, то есть $k = \log_2 |Q|$. Что и требовалось доказать.

Существует и обратная теорема.

Теорема 4.5.2. Всякая правильная синхронная логическая сеть (ПЛС) со входами x_1, \dots, x_m , выходами z_1, \dots, z_n и k элементами задержки является конечным автоматом, входной алфавит которого состоит из 2^m двоичных наборов длины m , выходной алфавит – из 2^n наборов длины n , а множество состояний – из 2^k наборов длины k .

Доказательство. Вначале возьмем ПЛС без задержек, а, следовательно, и без циклов. Такие сети носят название *линейно-упорядоченных сетей* (ЛУС), так как элементы такой сети можно пронумеровать таким образом, что любая межэлементная связь будет соединять выходной полюс элемента с меньшим номером с входным полюсом элемента с большим номером. Рассмотрим любой выход сети z . Блок, которому он принадлежит, реализует на этом выходе логическую функцию $z = f^1(p_1^1, p_2^1, \dots, p_r^1)$, где p_1^1, \dots, p_r^1 – входы блока. Каждый из этих входов является либо входом сети (переменной x), либо присоединен к выходу другого блока – $p_i^1 = f_i^2(p_{i_1}^2, p_{i_2}^2, \dots, p_{i_{m_k}}^2)$ и, таким образом,

$$z = f^1\left(f_1^2(p_{i_1}^2, p_{i_2}^2, \dots, p_{i_{k_1}}^2), f_2^2(p_{j_1}^2, p_{j_2}^2, \dots, p_{j_{k_2}}^2), \dots, f_r^2(p_{n_1}^2, p_{n_2}^2, \dots, p_{n_{k_r}}^2)\right),$$

где вместо некоторых f_i^2 стоят, возможно, переменные x . Следующее повторение этой процедуры дает формулу глубины 3 с функциями f_i^3 и переменными p_i^3 и т.д., причем верхний индекс у p_i^k равен числу блоков между полюсом p_i^k и выходом z . Так как сеть ациклическая, то этот процесс рано или поздно закончится, и все переменные будут заменены переменными x . Поскольку задержек в сети нет, то в результате получим $z(t)$

как логическую функцию от некоторых из $x_1(t), \dots, x_m(t)$ и, следовательно, ПЛС без задержек – это логический комбинированный автомат.

Теперь возьмем произвольную ПЛС (обозначим ее G). Удалив из нее элементы задержки, получим ЛУС G_0 без задержек, которая является логическим комбинационным автоматом. Входами G_0 являются: во-первых, входы G , а во-вторых, выходы z_1, \dots, z_k элементов задержки G , выходы G_0 – это выходы G и входы z_1', \dots, z_k' элементов задержки G (см. рис. 4.19). Таким образом, входной набор G_0 имеет вид $(x_1, \dots, x_m, z_1, \dots, z_k)$, а выходной набор – $(y_1, \dots, y_n, z_1', \dots, z_k')$. Теперь считаем набор $(x_1(t), \dots, x_m(t))$ входным сигналом $x(t)$ сети G , набор $(y_1(t), \dots, y_n(t))$ – выходным сигналом $y(t)$ сети G , а набор $(z_1(t), \dots, z_k(t))$ – состоянием $q(t)$ сети G . Учитывая, что $(z_1'(t), \dots, z_k'(t)) = (z_1(t+1), \dots, z_k(t+1)) = q(t+1)$, получим, что сеть G_0 вычисляет две системы логических функций от набора $x(t) \times q(t)$ – систему $(z_1'(t), \dots, z_k'(t)) = q(t+1)$, т.е. функцию переходов δ , и систему $(y_1(t), \dots, y_n(t))$, т.е. функцию выхода λ . Эти две системы функций называются *каноническими уравнениями сети G* .

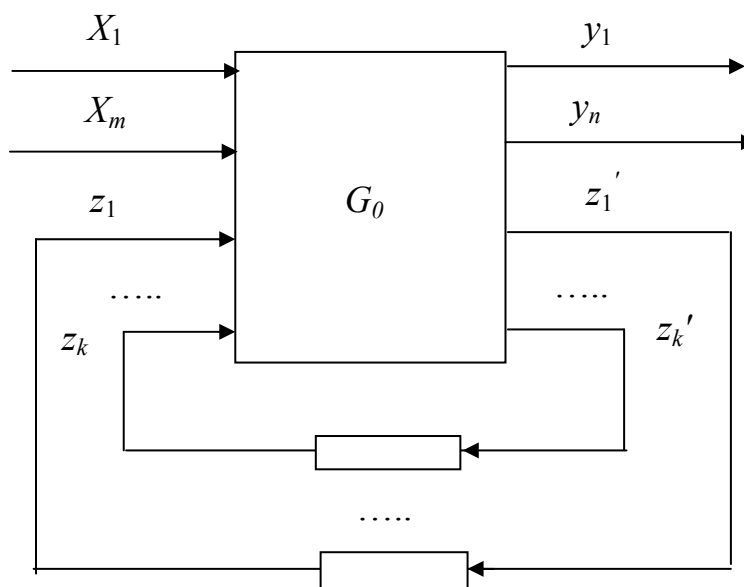


Рис. 4.19. Правильная синхронная сеть, представляющая автомат

Окончательно сеть G принимает вид рис. 4.19, где δ и λ образуют логический комбинационный автомат G_0 , а блок задержки D состоит из k двоичных задержек. Это и есть автоматное описание ПЛС. Что и требовалось доказать.

Рассмотрим теперь сети, составленные только из логических элементов и содержащие в отличие от ПЛС контуры (обратные связи). Анализируя поведение таких сетей, можно прийти к одному из выводов:

а) при любой комбинации значений входных полюсов сеть будет переходить в некоторое устойчивое состояние и оставаться в нем, пока входной сигнал не изменится;

б) при некоторых обстоятельствах условие предыдущего пункта может нарушаться, и в сети возникают противоречия (как, например, это было в сети, изображенной на рис. 4.17, а).

Сеть, удовлетворяющая первому выводу, называется *асинхронной сетью* и ее можно рассматривать как структуру асинхронного автомата.

Простейший пример приведен на рис. 4.20.

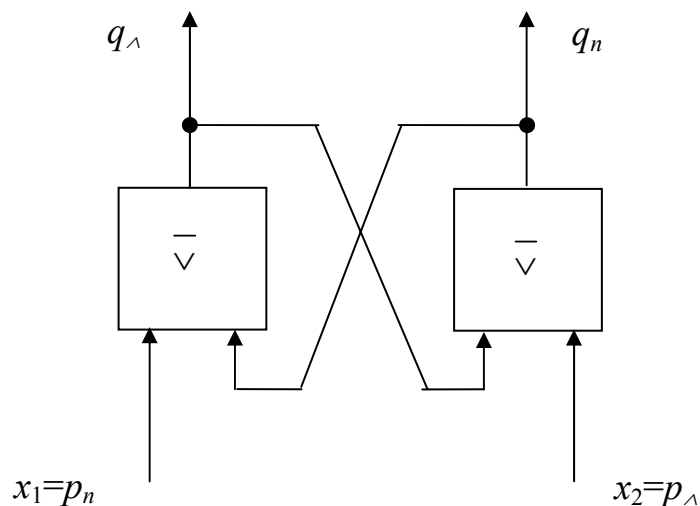


Рис. 4.20. Схема триггера

Легко видеть, что эта сеть оказывается триггером, который с учетом переобозначений входов полностью совпадает с изображенным на рис. 4.16.

Рассматривая триггеры как элементы, произвольный асинхронный автомат по аналогии с рис. 4.18 можно представить некоторым логическим комбинационным автоматом G_0 и совокупностью триггеров, концентрирующих в себе “память” автомата (рис 4.21). Вход и выход автомата представляются, как обычно, двоичными наборами x и y , а внутренние состояния – значениями вектора $q_n = (q_{1n}, q_{2n}, \dots, q_{kn})$, где q_{in} – правый выходной полюс i -го триггера. Переменная $q_{i\wedge}$ всегда (см. автоматную таблицу на рис.4.16) принимает инверсное значение по отношению к q_{in} , в связи с чем можно считать, что логические функции, реализуемые комбинационным автоматом G_0 , зависят только от переменных $x_1, \dots, x_m, q_{1n}, \dots, q_{kn}$.

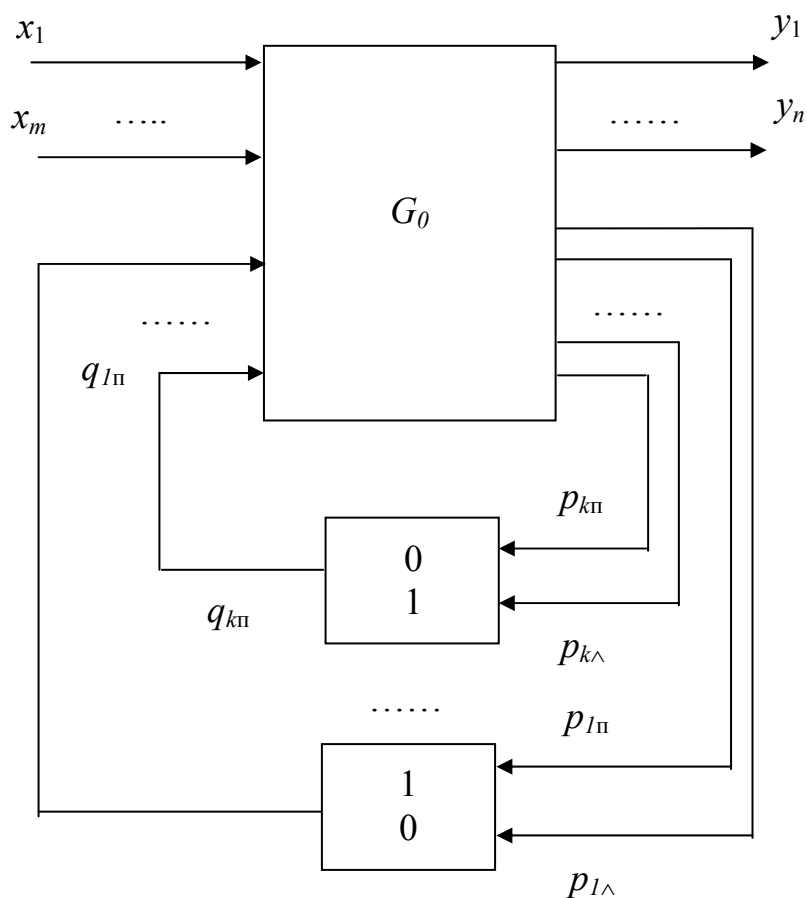


Рис. 4.21. Асинхронный автомат, представленный сетью

Эти функции представляют собой, во-первых, функции выхода:

$$y_i = \lambda_i(x_1, \dots, x_m, q_{1п}, \dots, q_{кп}), i = \{1, 2, \dots, n\},$$

а во-вторых, функции возбуждения триггеров:

$$p_{i\wedge} = \delta'(x_1, \dots, x_m, q_{1п}, \dots, q_{кп}), i = \{1, 2, \dots, k\}$$

(для левого входа), и

$$p_{iп} = \delta''(x_1, \dots, x_m, q_{1п}, \dots, q_{кп}), i = \{1, 2, \dots, k\}$$

(для правого входа).

При решении задачи синтеза асинхронного автомата большое значение имеет нахождение функций возбуждения триггеров. Определение этих функций допускает некоторую вольность, которая становится ясной при анализе автоматной таблицы триггера (рис. 4.16). Действительно, если при реализации некоторого перехода между внутренними состояниями автомата i -й триггер должен сменить состояние 0 на состояние 1 (условно обозначим этот факт как $0 \rightarrow 1$), то на его вход должна быть подана вполне определенная комбинация 01. Но если переход при смене состояний должен быть $0 \rightarrow 0$, то такое возможно как при комбинации на входе 00 (как не меняющей состояние триггера), так и при комбинации 10 (эта комбинация всегда переводит триггер в состояние 0 независимо от того, в каком состоянии он перед этим находится). Таким образом, на входе должна быть комбинация – 0, где прочерк означает произвольное значение (либо 0, либо 1). Анализируя автоматную таблицу триггера, придем к следующей таблице смены состояний триггера.

Таблица 4.4

Тип изменения состояний триггера	Требуемая комбинация на входе
$0 \rightarrow 0$	- 0
$0 \rightarrow 1$	01
$1 \rightarrow 0$	10
$1 \rightarrow 1$	0 -

Изображенный на рис. 4.20 триггер реализован на элементах Пирса, но возможна реализация триггера и на других логических элементах, например на элементах Шеффера. Из других видов триггеров полезно упомянуть счетный триггер (или триггер со счетным входом), таблица переходов которого приведена ниже.

Таблица 4.5

Тип изменения состояний	Значение входа
$0 \rightarrow 0$	0
$0 \rightarrow 1$	1
$1 \rightarrow 0$	1
$1 \rightarrow 1$	0

В состав счетного триггера входят два уже рассмотренных ранее триггера и логическая комбинационная схема. При небольших умственных затратах можно нарисовать структурную схему счетного триггера. Рекомендуется проделать это самостоятельно в качестве упражнения.

4.5.5. Анализ комбинационных автоматов

Из теорем 4.5.1 и 4.5.2. следует, что структуру комбинационного автомата всегда можно представить линейно-упорядоченной сетью, содержащей только логические элементы. Для краткости назовем такую сеть комбинационной.

Если соответствующее какой-либо комбинационной сети отображение $P \setminus X$ в Q является взаимно-однозначным отображением $P \setminus X$ на некоторое подмножество Q , то такая сеть называется *сходящейся*, если нет, то *расходящейся*. Необходимо напомнить, что P – множество входных полюсов элементов сети, X – множество входных полюсов сети, а Q – множество выходных полюсов элементов сети. То есть связи расходящейся сети

могут ветвиться (один выходной полюс может быть соединен с несколькими входными), а для сходящейся сети это невозможно.

Если комбинационная сеть имеет только один выходной полюс (реализует одну логическую функцию) и является сходящейся, то ее можно представить в виде одной формулы, задающей суперпозицию логических функций, реализуемых элементами сети. Для подобного представления расходящейся сети требуется уже система формул. Эта система может быть получена путем введения дополнительных переменных для обозначения тех связей, удаление которых превращает расходящуюся сеть в сходящуюся.

Ясно, что система формул требуется и для описания комбинационной сети с более чем одним выходным полюсом.

Исследование комбинационных автоматов сводится к исследованию отношений между логическими функциями и их представлением в виде суперпозиции элементарных функций, реализуемых отдельными элементами сети. При анализе по заданной суперпозиции, определяемой комбинационной сетью, находится формула (или система формул), представляющая логическую функцию (систему функций). Затем путем эквивалентных преобразований эта формула представляется в некоторой более удобной форме.

Пример 4.25. Проведем анализ сети, изображенной на рис. 4.22.

Вводя промежуточные переменные, соответствующие расходящимся выходам элементов, получаем систему формул:

$$e = b \vee c,$$

$$f = c \cdot d,$$

$$g = e \cdot f,$$

$$y = (ab \vee e)g \vee gf.$$

Подставляя в последнюю формулу выражения для промежуточных переменных и применяя правила эквивалентных преобразований булевых формул, получаем окончательно весьма простую булеву формулу $y=cd$, которую, очевидно, можно реализовать и одним элементом.

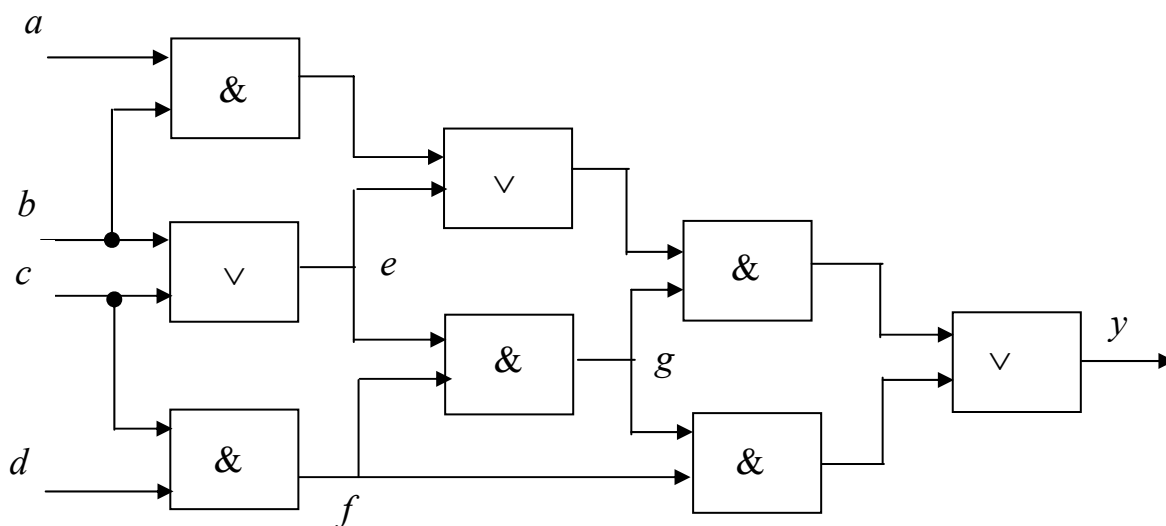


Рис. 4.22. К анализу сети (пример 4.25)

4.5.6. Синтез комбинационных автоматов

При синтезе заданную логическую функцию f необходимо представить в виде суперпозиции элементарных функций, определяющей структуру комбинационного автомата. Если функция f задана формулой F над множеством Σ элементарных функций, то формуле F можно поставить в соответствие схему G из логических элементов, реализующих функции Σ . Построение G осуществляется индукцией по глубине формулы:

1) если $F=\varphi(x_{i1},\dots x_{ik})$, где $\varphi\in\Sigma$, $x_{i1},\dots x_{ik}$ – исходные переменные, то схема G состоит из одного элемента φ , входы которого отождествляются с переменными $x_{i1},\dots x_{ik}$, а выход – с переменной y ;

2) если $F=\varphi(F_1,\dots F_k)$, где F_i – переменная x_{ji} или функция, уже реализованная схемой G_i , то схема G для F строится так: к i -му входу элемен-

та φ присоединяется выход схемы G_i (если F_i – функция) или входная переменная (если $F_i = x_{ij}$), выходом y будет выход элемента φ .

Такой способ построения всегда приводит к линейно-упорядоченной сходящейся сети, т. е. имеет форму дерева, причем входные полюса соответствуют концевым вершинам, а выход – корню дерева.

Понятно, что между множеством формул над Σ и множеством древовидных схем из элементов, реализующих функции Σ , существует взаимно-однозначное соответствие: по любой формуле F над Σ изложенный метод однозначно строит схему G и, наоборот, анализ схемы G (например, с помощью теоремы 4.5.2) дает исходную формулу F . Число знаков операций в F равно числу элементов схемы G . Все это сводит задачу преобразования схем (в том числе их минимизацию) к задаче преобразования логических формул.

Итак, по формуле над Σ всегда можно построить линейно - упорядоченную сходящуюся сеть из элементов Σ ; обратное утверждение в силу теоремы 4.5.2 верно для любой (не обязательно сходящейся) сети. Отсюда следует важный, хотя и очевидный факт: для того чтобы произвольная логическая функция могла быть реализована схемой над Σ , необходимо и достаточно, чтобы множество функций Σ было *функционально полным*.

Для системы функций справедливо то же самое.

Проблема минимизации схемных решений оказывается куда более сложной. Задача минимизации формул сама по себе сложна, но и она не исчерпывает возможности минимизации схем.

До настоящего времени известно очень небольшое количество классов функций, для которых найдены минимальные схемные решения. И в общем случае, видимо, поиск минимальных решений невозможен без большого перебора вариантов. Даже достаточно точно оценить по задан-

ной функции хотя бы число элементов в минимальной схеме (не проводя синтеза) также не удастся.

Из общих теоретических результатов здесь следует упомянуть теорему Шеннона – Лупанова.

Пусть $L_{\Sigma}(f)$ – число элементов минимальной схемы в базисе Σ , реализующей функцию f . Введем функцию $L_{\Sigma}(n) = \max_{f \in P_2(n)} L_{\Sigma}(f)$, где максимум берется по всем двоичным функциям от n переменных. Эта функция носит название функции Шеннона для базиса Σ и равна она минимальному числу элементов из Σ , достаточному для реализации любой функции n переменных.

Теорема Шеннона – Лупанова утверждает, что для любого базиса Σ

$$L_{\Sigma}(n) \approx C_{\Sigma} \frac{2^n}{n},$$

где C_{Σ} – константа, зависящая от базиса Σ , а знак \approx означает асимптотическое равенство.

При этом для любого $\varepsilon > 0$ доля функций f , для которых $L_{\Sigma}(f) \leq (1 - \varepsilon) C_{\Sigma} \frac{2^n}{n}$, стремится к нулю с ростом n .

Смысл второго утверждения теоремы в том, что с ростом n почти все функции реализуются со сложностью, близкой к верхней границе, т.е. к $L_{\Sigma}(n)$.

Ознакомимся теперь с методами синтеза, развитыми для конкретных базисов Σ .

Базис произвольных дизъюнктивных нормальных форм (ДНФ). Это базис, в котором синтезируются структуры, непосредственно соответствующие ДНФ. Элементами базиса являются конъюнкторы и дизъюнкторы с произвольным числом входов, реализующие конъюнкцию и дизъюнкцию любого числа переменных. Эти элементы могут соединяться так, чтобы

образовывались двухъярусные сети, в которых элементами первого яруса служат конъюнкторы, а элементами второго – дизъюнкторы. При этом выходные полюсы элементов первого яруса могут соединяться с входными полюсами элементов второго яруса, а выходные полюса элементов второго яруса являются выходными полюсами сети в целом. Входные же полюсы сети соединены непосредственно с входными полюсами элементов первого яруса. Эти правила соединения – синтаксис базиса.

Отмечая некоторые из входных полюсов сети символами инверсий аргументов, предполагаем, что эти инверсии получаются где-то вне синтезируемой сети и доступны измерению (наблюдению). Это обеспечивает функциональную полноту базиса.

Для реализации одной функции в базисе произвольных ДНФ нужно построить в данном базисе сеть с одним выходным полюсом. При этом сеть должна быть оптимальной в каком-то смысле. Например, если мы по ДНФ найдем известными методами *тупиковую* ДНФ, получим схему с минимальным числом элементов. Можно минимизировать число входных полюсов сети, отказываясь от дублирования прямых значений аргументов их инверсиями. Можно стремиться к минимуму инверсных значений. Можно заняться выравниванием нагрузки среди аргументов, когда минимизируется максимальное число конъюнкторов, непосредственно связанных с одним и тем же входным полюсом сети. Существуют и другие задачи, которые приводят к разным методам синтеза.

При синтезе комбинационного автомата, реализующего систему функций, возникают другие проблемы. Можно, конечно, реализовать каждую функцию отдельно, но такое решение вряд ли будет лучшим в каком-то смысле.

Пусть, например, требуется минимизировать число элементов в сети. Число элементов второго яруса, как правило, равно числу функций. Исключения бывают, когда функция представлена одной конъюнкцией

или когда некоторые функции равны либо становятся равными при соответствующем их дополнении. Это случаи тривиальные и поэтому неинтересные. Следовательно, основные усилия должны быть направлены на минимизацию числа элементов первого яруса.

Исходная же система функций может задаваться по-разному, в зависимости от таких параметров, как число функций, число аргументов, степень определенности функций, степень их взаимосвязи и т.д. Разными в таких случаях получаются и методы минимизации.

Функция Шеффера (элемент “И-НЕ” или инверсный конъюнктор) и *стрелка Пирса* (элемент “ИЛИ-НЕ” или инверсный дизъюнктор). Каждый из этих элементов может реализовать функционально полный базис.

Рассмотрим двухъярусные сети, элементами которых могут служить элементы Шеффера с произвольным числом входных полюсов.

Применим правило де Моргана к ДНФ простейшей функции, например, к $xy \vee zu$:

$$xy \vee zu = \overline{\overline{xy} \& \overline{zu}}.$$

Легко видеть, что синтез сети данного класса, реализующий заданную функцию, может быть сведен к синтезу соответствующей двухъярусной сети в базисе произвольных ДНФ с последующей заменой всех элементов полученной сети на элементы Шеффера.

К этому же можно свести и синтез двухъярусной сети на элементах Пирса. Для этого достаточно перейти от дизъюнктивной нормальной формы (ДНФ) к конъюнктивной нормальной форме (КНФ), построить соответствующую двухъярусную сеть (на этот раз первый ярус будет содержать дизъюнкторы, а второй – конъюнкторы) и согласно формуле

$$(x \vee y) \& (z \vee u) = \overline{\overline{x \vee y} \vee \overline{z \vee u}}$$

заменить все элементы построенной сети элементами Пирса.

Пример 4.26. Пусть логическая функция задана булевой формулой

$$f = (x_1 \vee x_2) x_3 \overline{x_4} (\overline{x_1} \vee x_3).$$

Синтезируем комбинационный автомат, реализующий данную функцию в *базисе ДНФ*. Для этого приведем заданную формулу к дизъюнктивной нормальной форме, т.е. к дизъюнкции элементарных конъюнкций, используя правила эквивалентных преобразований булевых формул:

$$f = (x_1 \vee x_2) x_3 \overline{x_4} (\overline{x_1} \vee x_3) = (x_1 \vee x_2) x_3 \overline{x_4} = x_1 x_3 \overline{x_4} \vee x_2 x_3 \overline{x_4}.$$

Затем строим двухъярусную сеть, в первом ярусе которой будет два конъюнктора, а во втором – один дизъюнктор (см. рис.4.23).

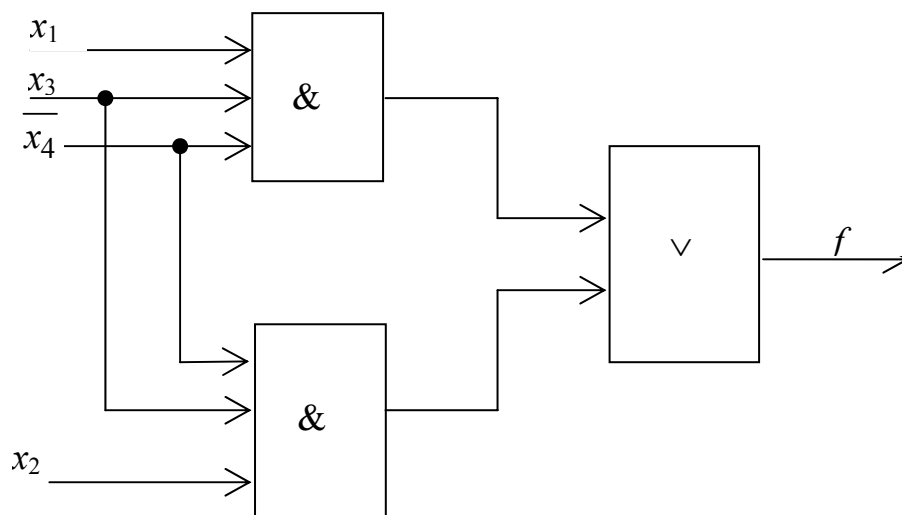


Рис.4.23. Сеть в базисе ДНФ

Если необходимо реализовать ту же логическую функцию в *базисе элементов Шеффера* (“И–НЕ”), то каждый элемент сети, представленной на рис. 4.23, заменяем элементом “И–НЕ” (см. рис.4.24).

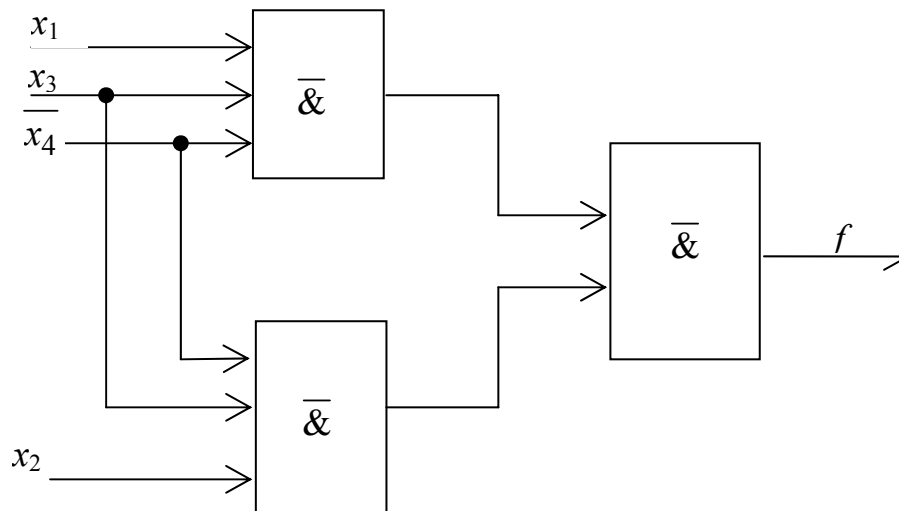


Рис.4.24. Сеть в базисе элементов Шеффера

Пример 4.27. Пусть логическая функция задана булевой формулой

$$f = \overline{x_1 x_2} \cdot (\overline{x_2} \vee x_1 x_3)$$

Для реализации данной функции в *базисе КНФ* необходимо вначале по формулам эквивалентных преобразований перейти к конъюнктивной нормальной форме, т.е. к конъюнкции элементарных дизъюнкций:

$$\begin{aligned} f &= \overline{x_1 x_2} \cdot (\overline{x_2} \vee x_1 x_3) = (\overline{x_1} \vee \overline{x_2}) (\overline{\overline{x_2} \vee x_1 x_3}) = (\overline{x_1} \vee \overline{x_2}) (\overline{x_2 \cdot x_1 x_3}) = \\ &= (\overline{x_1} \vee \overline{x_2}) (\overline{x_2 \cdot (x_1 \vee x_3)}) = (\overline{x_1} \vee \overline{x_2}) (\overline{x_2 x_1 \vee x_2 x_3}) = \\ &= (\overline{x_1} \vee \overline{x_2}) \overline{x_2 x_1} \cdot \overline{x_2 x_3} = (\overline{x_1} \vee \overline{x_2}) (\overline{x_2} \vee \overline{x_1}) (\overline{x_2} \vee \overline{x_3}) = (\overline{x_1} \vee \overline{x_2}) (\overline{x_2} \vee x_3). \end{aligned}$$

Сеть, реализующая данную формулу, представлена на рис. 4. 25.

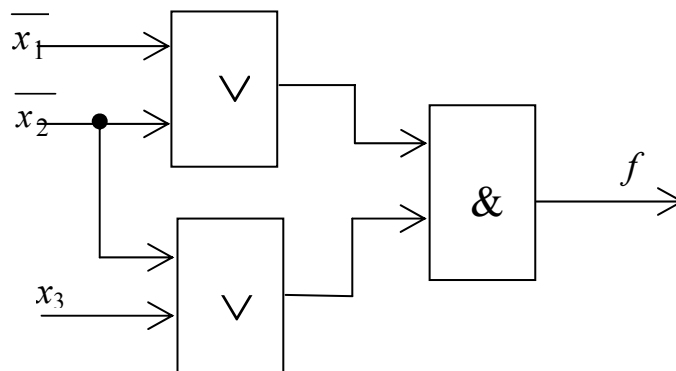


Рис. 4.25. Сеть в базисе КНФ

Для реализации той же функции в *базисе элементов Пирса* (“ИЛИ-НЕ”) заменяем каждый элемент сети на рис. 4.25 элементом “ИЛИ-НЕ” (см. рис. 4.26).

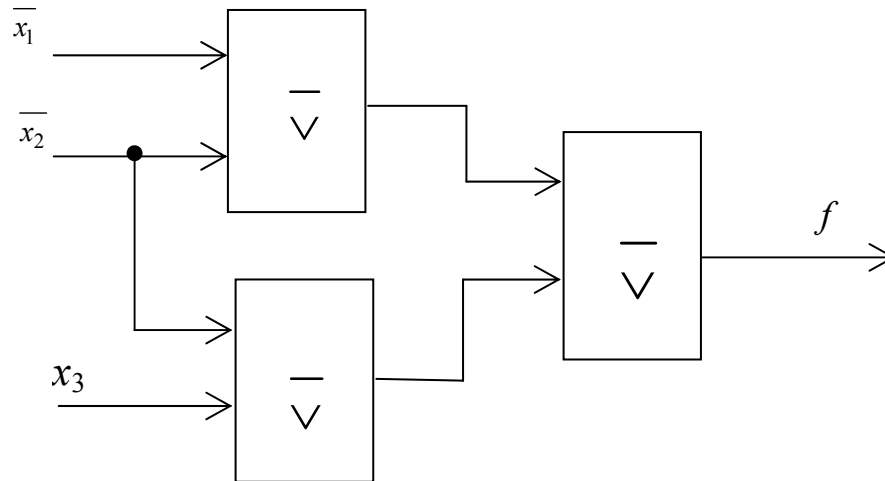


Рис. 4.26. Сеть в базисе элементов Пирса

4.5.7. Кодирование состояний

Речь пойдет о кодировании в основном внутренних состояний, так как двоичное кодирование входного и выходного алфавитов не вызывает принципиальных трудностей и практически не влияет на сложность полученных при этом структурных схем.

Пусть имеется автомат $A = (X, Q, Y, q_1 \in Q, F(x \in X / y \in Y))$ и набор элементов памяти U_1, U_2, \dots, U_p . Каждому состоянию q автомата A поставим в соответствие конечный упорядоченный набор (z_1, z_2, \dots, z_p) состояний автоматов U_1, \dots, U_p так, что различным состоянием автомата A ставятся в соответствие различные последовательности состояний элементарных автоматов U_1, \dots, U_p . Таким образом, состояния автомата A кодируются наборами состояний элементов памяти U_1, \dots, U_p , в результате чего возникают структурные состояния автомата A . Поскольку при практическом синтезе

схем используются в основном в качестве элементов памяти элементарные автоматы с двумя состояниями 0 и 1, то и состояния автомата A кодируется наборами двоичных переменных (z_1, \dots, z_p) (двоичное кодирование).

В зависимости от того, каким образом выполнять кодирование, структурные схемы одного и того же автомата могут получаться различными, так как каждому варианту кодирования соответствует структурная схема определенной сложности. Различные способы кодирования оказываются неравноценными как с точки зрения простоты структуры автомата, так и с точки зрения других критериев: быстродействия, надежности и прочее.

Самое главное, конечно, – это простота структуры автомата. В качестве промежуточной цели можно наметить простоту системы логических функций, реализуемых комбинационной частью автомата. Выбирая различные варианты кодирования, мы получаем различные системы логических функций. Можно минимизировать эти системы, например, в классе ДНФ, а затем подсчитывать число символов в полученных выражениях. Можно надеяться, что, выбирая способ кодирования, который приводит к выражениям с минимальным числом букв, мы получим в итоге и более простую структуру автомата.

В асинхронных автоматах могут возникать при кодировании другие проблемы, связанные с практической реализацией и конструктивными особенностями элементов памяти (триггеров). Каждый из реальных элементов памяти обладает инерционностью (ненулевое время срабатывания), причем эта инерционность не является постоянной и одинаковой для всех элементов. Это не учитывается в абстрактной модели автомата. Вследствие этого при переходе автомата из одного состояния в другое может реализоваться некоторая последовательность элементарных переходов (соответствующих изменениям состояния отдельных элементов памяти), при которой автомат проходит через некоторое множество промежуточных состояний и

которая в общем случае непредсказуема. Последующие действия автомата будут определяться уже значениями функции переходов на достигнутых промежуточных состояниях.

Таким образом, дальнейшее поведение автомата может оказаться в зависимости от того, какой из элементов памяти быстрее среагирует на прикладываемое к нему воздействие. Элементы как бы состязаются в быстроте реакции, чем и обусловлено название соответствующего явления — состязания между элементами памяти. Если, в конце концов, автомат приходит в намеченное матрицей переходов состояние, то состязания можно считать неопасными, в противном случае их следует рассматривать как опасные.

Чтобы поведение автомата не отличалось от заданного матрицей переходов, необходимо устранить все опасные состязания между элементами памяти. Один из возможных способов следующий.

Заданный в автомате переход $i \rightarrow j$ можно обеспечить, если придать функции переходов значение j на всех возможных промежуточных состояниях, в которые автомат может попасть из состояния i (при фиксированном входном состоянии). В этом случае элементы памяти, которые должны менять свои состояния, будут подвергаться постоянному надлежащему воздействию на всем протяжении рассмотренного перехода независимо от того, в каком порядке они срабатывают. Тогда состязания становятся неопасными, и неизбежно наступит переход $i \rightarrow j$, который назовем *прямым*, причем происходить он будет с максимальным быстродействием.

Одним из эффективных способов предотвратить опасные состязания является рациональное кодирование внутренних состояний автомата. Все такие методы можно условно разбить на два класса. В одном из них ищутся такие коды, при которых все заданные переходы становятся элементарными (меняет состояние только один элемент памяти) и, следовательно, состязаний вообще нет. Если для исходной матрицы переходов та-

кое сделать не удастся, то матрица преобразуется, причем множество состояний, как правило, расширяется. Во втором классе находятся методы, устраняющие лишь опасные состязания и не связанные с увеличением числа внутренних состояний. Эти методы основаны на реализации прямых переходов.

В качестве примера рассмотрим один из методов подобного типа.

Вначале необходимо сформулировать необходимые и достаточные условия отсутствия опасных состязаний.

Пусть $U(i, j)$ – множество всех возможных промежуточных состояний, включая состояния i и j , в которые автомат может попасть при переходе $i \rightarrow j$. По определению функция переходов – однозначная функция полного состояния, а при фиксированном входном состоянии – однозначная функция внутреннего состояния. Отсюда следует, что прямые переходы могут быть реализованы тогда и только тогда, когда выполняется условие: для каждой пары переходов $i \rightarrow j$ и $k \rightarrow l$, соответствующих одному и тому же входному состоянию, множества $U(i, j)$ и $U(k, l)$ не пересекаются, т.е. $U(i, j) \cap U(k, l) = \emptyset$, если $j \neq l$.

Найти множество $U(i, j)$ можно, зная коды состояний i и j . Пусть эти коды будут $z(i)$ и $z(j)$. Множество $U(i, j)$ будет образовано всеми теми состояниями, коды которых совпадают с кодами $z(i)$ и $z(j)$ в тех компонентах, которые совпадают между собой в векторах $z(i)$ и $z(j)$. Таким образом, множество $U(i, j)$ (точнее множество соответствующих кодов) может быть представлено вектором $t(i, j)$, компоненты которого принимают значения одноименных компонентов векторов $z(i)$ и $z(j)$ в случае совпадения последних и значение “–” в противном случае. Коды всех элементов множества $U(i, j)$ могут быть получены из $t(i, j)$ подстановкой вместо прочерков нулей и единиц.

Необходимым и достаточным условием непересечения множеств $U(i, j)$ и $U(k, l)$ является наличие такой одноименной компоненты в кодах $t(i, j)$ и $t(k, l)$, которая принимает значение 1 в одном и 0 в другом коде. Доказательство этого утверждения очевидно; если это условие выполнено, то любой элемент множества $U(i, j)$ отличается от любого элемента множества $U(k, l)$ (именно этой компонентой), в противном случае можно в этих множествах найти общий элемент.

Кодирующей матрицей назовем такую матрицу Q с двоичными элементами, столбцами которой будут коды внутренних состояний, а строкам поставлены во взаимно-однозначное соответствие внутренние состояния автомата. Получение такой матрицы и есть цель кодирования.

Для матрицы Q необходимое и достаточное условие отсутствия опасных состязаний можно сформулировать так: для каждой пары $i \rightarrow j$ и $k \rightarrow l$, соответствующих одному и тому же входному состоянию ($j \neq l$), в матрице Q должна найтись строка, в которой i -я и j -я компонента принимают значение 1 (или 0), а k -я и l -я компоненты – значение 0 (или 1).

Условие, предъявляемое при этом к матрице Q при рассмотрении конкретной пары переходов $i \rightarrow j$ и $k \rightarrow l$, назовем *элементарным*. Оно может быть представлено вектором, у которого i -я и j -я компоненты инверсны по отношению к k -й и l -й компонентам, а остальные компоненты – прочерки (длина такого вектора равна числу внутренних состояний автомата). Совокупность таких векторов для всех пар и всех входных состояний образует матрицу условий S .

Задача нахождения матрицы Q , удовлетворяющей совокупности условий, задаваемых матрицей S , сводится к задаче нахождения минимальной покрывающей формы для матрицы S .

Рассмотрим последнее утверждение. Можно считать, что каждая из строк матрицы S задает некоторое частичное двухблочное разбиение на

множестве столбцов матрицы Q . Один блок – это столбцы, отмеченные единицей (которая стоит в соответствующих компонентах строки матрицы условий), другой блок – это столбцы, отмеченные нулями. Прочерк, как и ранее, является признаком неопределенности – эти столбцы могут принадлежать любому блоку. Таким образом, решаемая задача сводится к нахождению такой кодирующей матрицы Q , которая бы покрывала матрицу условий S . При этом опасные состязания будут отсутствовать. Дополнительно хотелось бы, чтобы число строк такой матрицы было бы минимальным, что соответствует минимальному количеству элементов памяти.

Таким образом, всю процедуру нахождения кодирующей матрицы Q можно сформулировать следующим образом.

1. Считаем, что все состояния автомата различны, т. е. их надо кодировать разными кодами. Отсутствие эквивалентных состояний может гарантировать предварительная минимизация автомата на абстрактном уровне или добавление в матрицу переходов лишнего столбца, значения элементов которого совпадают с номерами строк, где они находятся.

2. Составляем матрицу условий S для всех входных состояний, выбрасывая строки, которые покрываются другими строками. В результате получаем минимальную матрицу условий S_0 .

3. Известными методами находим минимальное покрытие полученной матрицы S_0 . Найденная матрица и будет минимальной кодирующей матрицей. Число строк этой матрицы равно необходимому минимальному числу элементов памяти автомата.

Необходимо понимать, что при описанном кодировании устраняются только опасные состязания, но не гарантируется оптимальность полученного решения в смысле минимума памяти.

4.5.8. Программная реализация комбинационных автоматов

Комбинационный автомат вычисляет некоторую логическую функцию или систему функций, а, как известно, любой процесс вычисления может быть реализован как в аппаратном виде (схема из элементов), так и программным образом.

Под программой будем понимать некоторую пронумерованную последовательность команд k_1, \dots, k_s , взятых из некоторого фиксированного набора (системы команд). Программа работает над конечным множеством пронумерованных (поименованных) двоичных ячеек. Номер ячейки – это ее адрес. Адресом ячейки может служить и имя логической переменной, значения которой хранятся в данной ячейке. Система команд содержит команды – операторы типа $b := f(a_1, \dots, a_p)$ – выполнить операцию f над содержимым ячеек a_1, \dots, a_p и записать результат в ячейку b ; и условные двухадресные переходы двух видов:

1) “если a , то i , иначе j ” (если $a=1$, то перейти к выполнению команды k_i , иначе перейти к команде k_j) и

2) “если \bar{a} ($a=0$), то i , иначе j ”. Операция $f(a_1, \dots, a_p)$ – это логическая функция p переменных (в частном случае она может быть константой 0 или единицей). Если j – это номер следующей по порядку команды, то переход можно считать одноадресным, если $i=j$ – то это безусловный переход. Любая из перечисленных команд может быть заключительной, что указывается словом “конец”.

Процессом вычисления программы k_1, \dots, k_s называется последовательность шагов $k(1), \dots, k(t)$, на каждом из которых выполняется одна команда программы. Указанная последовательность определяется так:

1) $k(1) = k_1$,

- 2) если $k(i) = k_r$ – оператор, то $k(i+1) = k_{r+1}$;
- 3) если $k(i)$ – условный переход, то номер команды $k(i+1)$ указывается этим переходом;
- 4) если $k(i)$ – заключительная команда, то процесс вычисления останавливается после ее выполнения.

Программа Π вычисляет некоторую логическую функцию $y=f(x_1, \dots, x_n)$, если для любого двоичного набора $\sigma=(\sigma_1, \dots, \sigma_n)$ при начальном состоянии $x_1=\sigma_1, x_2=\sigma_2, \dots, x_n=\sigma_n$ программа через конечное число шагов останавливается и при этом в ячейке y будет величина $f(\sigma_1, \dots, \sigma_n)$.

Критерии, по которым можно оптимизировать программу, следующие:

- 1) число команд в тексте программы;
- 2) объем промежуточной памяти, то есть число ячеек, необходимых для хранения промежуточных результатов;

3) время вычисления – среднее $t_{cp}(\Pi) = \frac{1}{2^n} \sum_{\sigma} \tau_n(\sigma)$ и максимальное – $t_{max}(\Pi) = \max_{\sigma} \tau_n(\sigma)$, где $\tau_n(\sigma)$ – время работы программы на конкретном наборе σ , а сумма и максимум берется по всем 2^n наборам.

Любой линейно-упорядоченной сети (а следовательно, и любому комбинационному логическому автомату), содержащей N элементов и реализующей функцию f , нетрудно поставить в соответствие программу, вычисляющую f и состоящую из N команд, следующим образом. Занумеруем элементы сети числами $1, \dots, N$ в соответствии с линейной упорядоченностью. Номер 1 получит один из входных элементов, номер N – выходной элемент.

Пусть элемент e_i реализует функцию ϕ_i и к его входным полюсам присоединены выходные полюсы элементов e_{j1}, \dots, e_{jp} . Некоторые из них возможно являются входными полюсами сети. Поставим в соответствие

элементу e_i ячейку a_i и команду $a_i := \varphi_i(a_{j1}, \dots, a_{jp})$, если $i \neq N$ или ячейку y и команду $y := \varphi_N(a_{j1}, \dots, a_{jp})$ конец, если $i = N$. Получим в результате программу, не содержащую условных переходов (так называемая *операторная* программа), в которой порядок команд в точности соответствует порядку элементов в сети, а система команд – базису сети.

Проблема синтеза операторных программ сводится в основном к проблемам синтеза комбинационных сетей: в частности, задачи функциональной полноты системы команд и минимизации собственно текста программы соответствуют задачам о функциональной полноте системы функций и минимизации комбинационных схем. Так как операторные программы не содержат условных переходов, время ее вычисления на любом наборе одинаково и совпадает с максимальным временем $t_{cp} = t_{max} = N$ и в силу теоремы Шеннона – Лупанова при больших n приближается к $\frac{2^n}{n}$. А проблема минимизации памяти (за счет многократного использования одной и той же ячейки для нескольких последовательно получающихся промежуточных результатов) для таких программ – нетривиальная комбинационная задача.

Другой вид программ – это программы, состоящие из команд типа $y := \sigma$ ($\sigma = 0$ или $\sigma = 1$) и условных переходов. Такие программы называются *бинарными*. Всякую булеву формулу F , содержащую N символов, можно реализовать бинарной программой, вычисляющей F за время $t_{max} = N$ и содержащую N команд условного перехода, а также две команды $y = 0$ и $y = 1$ (эти команды не вошли в общее время t_{max}). Чтобы было понятнее, представим программу в виде графа, где вершины – это команды, а ребра – переходы. Пусть G_1 – граф программы для функции f_1 с начальной вершиной V_{10} и двумя заключительными вершинами V_{1z}^0 (с командой $y = 0$) и V_{1z}^1 (с командой $y = 1$), а G_2 – граф программы, реализующей функцию f_2 с начальной V_{20} и заключительными V_{2z}^0 и V_{2z}^1 вершинами. Тогда:

а) вычислять функцию $f=f_1 \vee f_2$ будет программа, граф G которой получен присоединением G_2 к “нулю” G_1 (то есть отождествлением вершин V_{1z}^0 и V_{20} ; команда $y:=0$ при этом отбрасывается);

б) вычислять функцию $f=f_1 \& f_2$ будет программа, граф G' которой получен присоединением G_2 к “единице” G_1 (отождествлением V_{1z}^1 и V_{20});

в) вычислять отрицание $f=\overline{f_1}$ будет программа, граф которой получен из G_1 заменой команд в V_{1z}^0 и V_{1z}^1 на инверсные.

В графе G (пункт а) получаются при этом две единичные, а в графе G' (пункт б) две нулевые вершины. В обоих случаях их надо отождествлять.

Пример 4.27. Граф бинарной программы, реализующей булеву формулу $f = (x_1 \vee x_2) x_3 \overline{x_4}$, приведен на рис. 4.27.

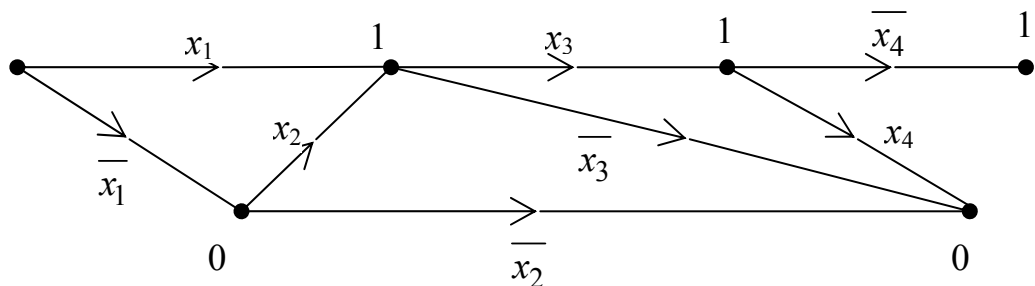


Рис. 4.27. Граф бинарной программы

Рассмотренный метод не гарантирует оптимальность получаемых программ в смысле минимума времени или минимума числа команд. Существуют и другие методы.

Показатели качества бинарных программ характеризуются следующими параметрами: $L_\delta(n)$ – функция Шеннона для числа команд бинарных программ, асимптотически равна $\frac{2^n}{n}$, причем существуют методы

синтеза, для которых $t_{\max} \sim n$.

Доля функций (для любого $\varepsilon > 0$), для которых $L_{\bar{6}}(f) \leq (1 - \varepsilon) \frac{2^n}{n}$ и $t_{\text{cp}}(f) \leq (1 - \varepsilon) \cdot n$, стремится к нулю с ростом n .

То есть сложность бинарных программ по числу команд асимптотически равна сложности операторных программ, но в отличие от операторных программ бинарные имеют два преимущества – это отсутствие промежуточной памяти и более высокое быстродействие, которое можно охарактеризовать соотношением

$$(\log_2 n + 1) \leq t_{\max}(f) \leq n.$$

Если в программе использовать и операторы и условные переходы, то число команд, асимптотически равное для операторных и бинарных программ $\frac{2^n}{n}$, можно понизить вдвое.

4.6. Общие методы синтеза автоматов

Для проведения синтеза автоматов нужно уметь представлять любой автомат в виде некоторой совокупности более простых автоматов. В разделе 4.5 было рассмотрено представление автоматов в виде сетей, то есть в виде соединения элементарных автоматов. Как это сделать практически и не обязательно в виде элементарных автоматов, а в общем случае в виде произвольных либо стандартных автоматов? Эти задачи решаются различными методами декомпозиции автоматов.

4.6.1. Декомпозиция абстрактных автоматов

Под декомпозицией в общем случае понимается представление сложного автомата работой нескольких более простых (в частном случае элементарных) автоматов, которые на структурном уровне с помощью отождествления входных и выходных полюсов образуют функциональную или структурную схему сложного автомата. Обычно ставится задача оптимальной декомпозиции с точки зрения минимального числа элементарных автоматов. В результате оптимальной декомпозиции осуществляется минимизация числа логических элементов, входящих в комбинационную часть.

На абстрактном уровне декомпозиция сложного автомата соответствует параллельной, последовательной или смешанной работе более простых автоматов, т.е. сводится к задаче разложения автомата по операции умножения, суммирования, суперпозиции или по нескольким операциям сразу. Поэтому можно рассматривать декомпозицию параллельную, последовательную или смешанную.

Параллельная декомпозиция соответствует разложению автомата в произведение или сумму двух или большего числа абстрактных автоматов, каждый из которых проще, чем исходный автомат. Здесь можно говорить о параллельной одновременной (умножении) или параллельной поочередной (сумме) декомпозиции автоматов. Последовательная декомпозиция соответствует разложению автомата по операции суперпозиции, а смешанная – одновременно по двум операциям (например умножения и суперпозиции).

Все описанные случаи декомпозиции – это “чистые” случаи декомпозиции автоматов. Таких автоматов, которые бы раскладывались только в параллельную или в последовательную или даже в смешанную работу автоматов, довольно мало по сравнению с теми, которые не раскладываются таким образом. Поэтому вводится понятие *общей декомпозиции* автомата,

которая понимается как совместная работа элементарных автоматов со связями между ними. Общая декомпозиция соответствует разложению абстрактного автомата в композицию двух или более элементарных автоматов, то есть соответствует представлению сложного автомата в виде сети из более простых автоматов. Таким образом, последовательная, параллельная или смешанная декомпозиция могут рассматриваться как частные случаи общей декомпозиции автоматов.

Необходимо заметить, что при разложении автомата по операции композиции ставится, как правило, задача *оптимальной* декомпозиции, то есть представление произвольного автомата совместной работой элементарных автоматов с *минимальным* числом связей между ними. Решение задачи оптимальной декомпозиции приводит к минимальной комбинационной части автомата.

Следующий уровень этой же задачи связан с реализацией автомата в однородных вычислительных средах и заключается в декомпозиции автомата на *заданные* автоматы, например, на элементарные автоматы из некоторого элементного базиса.

Свойство автомата быть представленным параллельной или последовательной работой более простых автоматов вытекает из анализа вида матриц, получаемых в результате произведения, суммирования или суперпозиции автоматов.

Нет необходимости здесь рассматривать все теоремы, касающиеся данного раздела. Из основных результатов можно назвать следующий: любое параллельное соединение автоматов (параллельное одновременное, с общим входом, не одновременное) можно представить в виде последовательного соединения (возможно, других автоматов), то есть в виде суперпозиции автоматов.

Если же мы хотим выделить стандартные автоматы из исходного автомата, то использование алгебраических операций позволяет на абст-

рактном уровне решить задачу декомпозиции сложного автомата на заданные стандартные автоматы путем сведения ее к решению суперпозиционных уравнений над автоматами.

Что касается общей декомпозиции автоматов, то здесь можно упомянуть следующее утверждение: любой автомат Мили с числом состояний $n > 2$ можно представить совместной работой (композицией) двух или большего числа простых автоматов, один из которых – автомат Мили, а остальные – автоматы Мура.

4.6.2. Канонический метод синтеза

Вначале подведем некоторый итог по поводу функциональной полноты элементного базиса синтеза автоматов. Автоматно полный базис согласно теоремам о представлении автомата соответствующей сетью (теорема 4.5.1) и схемной реализации логических функций (теорема 4.5.2) может представлять собой элемент единичной задержки и какую-либо функциональную полную систему логических элементов. Вместо элемента задержки в автоматный базис можно включить любой другой элемент памяти (например триггер). В качестве элемента памяти можно взять и любой автомат Мура с произвольным числом внутренних состояний (два, три, пять и т.д.), лишь бы этот автомат удовлетворял требованиям полноты системы переходов и полноты системы выходов.

Требование полноты системы переходов предусматривает для любой упорядоченной пары состояний элементарного автомата наличие некоторого входного сигнала, который переводит первый элемент этой пары во второй.

Требование полноты системы выходов означает, что для каждого состояния элементарного автомата имеется соответствующий ему выход-

ной сигнал, который отличается от выходного сигнала, соответствующего другим состояниям элементарного автомата.

Для тех элементов памяти, которые были рассмотрены, эти требования выполняются.

Доказано утверждение о том, что всякая система элементарных автоматов, которая содержит автомат Мура с нетривиальной памятью и полной системой переходов и выходов и функционально полную систему логических элементов, является автоматически полным базисом.

Но, к сожалению, в общем случае для произвольного набора элементарных автоматов проблема автоматной полноты оказывается *алгоритмически неразрешимой* (в отличие от подобной проблемы для комбинационного автомата).

Весь процесс синтеза можно разбить условно на ряд этапов. Классической считается схема синтеза, называемая *канонической схемой*, или *каноническим методом* синтеза, на разных этапах которой производятся следующие действия:

- 1) по описанию автоматного отображения (например, по регулярному событию) строится абстрактный автомат;
- 2) минимизируется число состояний автомата;
- 3) производится двоичное кодирование входного, внутреннего и выходного алфавитов (с учетом соображений раздела 4.5.6);
- 4) осуществляется выбор типов элементарных автоматов и определение их функций возбуждения в соответствии с кодированными автоматными таблицами переходов элементарных автоматов;
- 5) находятся минимальные формы для функций возбуждения;
- 6) получают с дальнейшей минимизацией функции выхода элементарных автоматов;
- 7) составляются канонические уравнения, описывающие комбинационные блоки автомата;

8) производится реализация системы канонических уравнений системой логических элементов в функционально полном базисе с последующей минимизацией.

Эта схема сыграла большую роль в развитии методов синтеза, но в практическом применении не очень удобна. Дело в том, что, во-первых, схемы с k элементами памяти имеют 2^k состояний, поэтому описание сколько-нибудь больших схем в терминах состояний и таблиц переходов получаются очень громоздкими. Во-вторых, на разных этапах решаются разные задачи минимизации, порой не только не связанные между собой, но и противоречащие друг другу. Например, известны примеры, когда уменьшение числа состояний приводит к усложнению комбинационной части. И в-третьих, различное кодирование (а для k элементов памяти вариантов таких кодов будет $(2^k)!$) приводит, вообще говоря, к разным системам канонических уравнений, сложность которых до начала синтеза предвидеть невозможно. Поэтому получают распространения другие методы синтеза, например, декомпозиционный.

4.6.3. Декомпозиционный метод синтеза

Этот метод основан на общей декомпозиции автоматов (обычно проводят оптимальную декомпозицию) на элементарные автоматы, в качестве которых могут быть выбраны любые абстрактные автоматы с простым числом состояний, например с двумя состояниями.

В результате декомпозиции получают матрицы соединений абстрактных элементарных автоматов, совместная работа которых эквивалентна работе исходного автомата. Выбирая конкретные типы элементарных автоматов (триггеры, элементы задержки и т.п.), по найденным матрицам соединения абстрактных элементарных автоматов строят функции возбуждения и функции выходов конкретных выбранных элементарных автома-

тов. А это, в свою очередь, определяет структурную схему комбинационной части и, следовательно, всего автомата в целом. Поэтому в этом методе, по сути, синтез на структурном уровне выносится на абстрактный уровень и сводится он к оптимальной в смысле минимума связей декомпозиции автомата на абстрактные элементарные автоматы (как правило, с двумя состояниями) и записи функций возбуждения и выходов конкретных элементарных автоматов по матрицам соединений абстрактных элементарных автоматов.

Преимущества декомпозиционного метода синтеза автоматов по сравнению с каноническим следующие:

- 1) не требуется строить сложные кодированные таблицы переходов;
- 2) решается проблема оптимального кодирования внутренних состояний автомата, приводящего к минимальной комбинационной части автомата;
- 3) декомпозиционный метод позволяет строить оптимальную или близкую к ней функциональную схему автомата при использовании элементарных автоматов со многими устойчивыми состояниями и логическими элементами в недвоичной логике. Таким образом, этот метод позволяет осуществлять синтез автоматов при использовании элементного базиса, использующего логику более высокого порядка по сравнению с двоичной.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. *Перегудов Ф.И., Тарасенко Ф.П.*, Основы системного анализа. – Томск: Изд-во НТЛ, 2003. – 396 с.
2. *Вуниш Г.* Теория систем. – М.: Советское радио, 1978. – 288 с.
3. *Кузнецов О.П., Адельсон-Вельский Г.М.* Дискретная математика для инженера. – М.: Энергоатомиздат, 1988. – 480 с.
4. *Мелихов А.Н.* Ориентированные графы и конечные автоматы. – М.: Наука, 1971. – 416 с.
5. *Закревский А.Д.* Алгоритмы синтеза дискретных автоматов. – М.: Наука, 1971. – 512 с.

Для заметок

Для заметок

Александр Георгиевич Карпов

Математические основы теории систем
Часть 1

Учебное пособие

Редактор Т.С. Портнова
Верстка Д.А. Звонков

К-ОКП ОК-005-93, код продукции 954240

Изд. лиц. ИД №04000 от 12.02.01. Подписано к печати 28.11.07.
Формат 60×84¹/₁₆. Бумага офсетная. Печать офсетная. Гарнитура «Times».
Усл. п. л. 10,7. Уч.-изд. л. 11,98. Тираж 100 экз.

ООО «Издательство научно-технической литературы»
634050, г. Томск, пр. Ленина, 34а, тел. (382-2) 53-33-35

Отпечатано Томск. гос. ун-т систем управления и радиоэлектроники, г. Томск,
пр. Ленина, 40