МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(национальный исследовательский университет)

П.П. Порешин, Б.Н. Попов

ДИСКРЕТНАЯ МАТЕМАТИКА:

МНОЖЕСТВА, ОТНОШЕНИЯ, ЛОГИКА, АВТОМАТЫ

*Рекомендовано Учебно-методическим объединением
по образованию в области прикладной информатики
в качестве учебного пособия для студентов
высших учебных заведений, обучающихся по направлению
«Прикладная информатика (по областям)»*

Москва
Издательство МАИ-ПРИНТ
2014

УДК 510.6-681.5
ББК 22.176
П 59

П 59 **Порешин П.П., Попов Б.Н.**

Дискретная математика: множества, отношения, логика, автоматы: Учебное пособие – М.: Изд-во МАИ-ПРИНТ, 2014. – 188 с.: ил.

ISBN 978-5-7035-2325-4

В учебном пособии в доступной форме изложены основные разделы дискретной математики: множества и отношения, логика и автоматы. Рассматриваются базовые объекты дискретной математики, свойства и методы их применения для решения практических задач. Теоретический материал: определения, теоретические результаты, модели и методы – излагается в виде, удобном для инженерной практики. Большое число задач снабжены решениями. В конце каждого раздела приведен список задач для самостоятельной проработки.

Для студентов, обучающихся по направлению «Системы управления летательными аппаратами».

Рецензенты:

кафедра «Высшая математика» МГТУ МИРЭА (зав. кафедрой д-р техн. наук, проф. *Ю.О. Худак*);

д-р техн. наук, проф. кафедры «Кибернетика» НИЯУ МИФИ *Б.А. Щукин*

ISBN 978-5-7035-2325-4

ОГЛАВЛЕНИЕ

ПРЕДИСЛОВИЕ

Учебное пособие обеспечивает курс «Дискретная математика» и соответствует разделам ряда других учебных дисциплин в рамках подготовки дипломированных специалистов, обучающихся по специальности 161101 «Системы управления летательными аппаратами». Кроме помощи в самостоятельном изучении материала, учебное пособие предназначено для выполнения УИРС, курсового и дипломного проектирования. Оно может быть полезным для молодых специалистов данного профиля деятельности и аспирантов.

Дискретная математика находит все более широкое применение в практике создания бортовых систем управления (БСУ) беспилотными космическими и атмосферными летательными аппаратами. Так, при разработке бортовых цифровых вычислительных систем используются математическая логика и способы минимизации логических функций. Комплексирование аппаратурного состава базируется на теории множеств. Формирование циклограмм полета предполагает графическое задание отношений. Существенным фактором при разработке бортового программного обеспечения, особенно для космических аппаратов, имеющих длительные сроки активного существования, является анализ нештатных ситуаций (НШС) и принятие решений по выходу из этих ситуаций. Теоретической основой отработки НШС могут служить теория конечных автоматов и логика предикатов.

В отличие от многочисленных изданий, посвященных различным разделам дискретной математики, в учебном пособии сделана попытка соединить такие разделы, как теория множеств, отношения, математическая логика, полные системы логических функций, минимизация и теория конечных автоматов в единый взаимосвязанный материал. Этот материал должен послужить теоретической основой курсов лекций по техническому, алгоритмическому и программному обеспечению БСУ.

В пособии большое внимание уделено практическим задачам и примерам.

Базовым материалом послужили лекции, прочитанные авторами для студентов МИФИ и МАИ в течение ряда лет.

Авторы благодарят сотрудников МОКБ «Марс» Качалову Е.Э. и Кособокову Т.В. за помощь в подготовке рукописи.

Учебное пособие подготовлено П.П. Порешиным (разд. 1, 2, 3, 4.1, 4.2.1, 4.3, 5, 6, 7, 8) и Б.Н. Поповым (предисловие, разд. 4.2.2, 4.4).

1. МНОЖЕСТВА

1.1. Основные понятия теории множеств

Понятие «***множества***» относится к фундаментальным понятиям в математике. И, как следствие этого, оно не имеет строгого, конструктивного определения. В теории вообще, и в теории множеств в частности, отсутствие такого определения компенсируется наличием ***интуитивного*** (субъективного, т.е. ориентированного на умственные способности) определения множества. Интуитивное определение любого понятия теории обеспечивает субъект руководящими правилами, посредством которых можно отделять объекты, удовлетворяющие определению от остальных объектов. И такие правила (свойства) для понятия множества были сформулированы одним из основоположников теории множеств – Георгом Кантором (1845–1918).

> ***Множество*** – это совокупность неких объектов, элементов, обладающих свойствами *общности* и *различия*.

Свойство *общности* позволяет объединять, группировать некие объекты в нечто единое целое, что и называют множеством. Математики договорились фиксировать это свойство посредством фигурных скобок – $\{\}$.

Свойство *различия* позволяет различать, отличать, отделять один объект (элемент) от другого внутри такой общности. Свойство различия обычно фиксируется запятыми внутри фигурных скобок $\{ , , , \}$, которые отделяют один элемент множества от другого. Из определения этих свойств следует, что множество не содержит ни одной пары идентичных, неразличимых элементов. Введенное определение не снимает проблемы поиска конструктивного определения понятия множества. Более того, интуитивное определение способно порождать противоречия, коллизии, характерные для математических теорий, построенных на его основе.

Далее по тексту *ПРОПИСНЫЕ* буквы используются для обозначения множеств, а *строчные* латинские буквы – для обозначения элементов множества. Запись $a \in A$ означает, что элемент a принадлежит множеству A. Запись $b \notin A$ означает, что элемент b не принадлежит множеству A.

В теории множеств не накладываются никакие ограничения на элементы множества. Множество может содержать различные объекты. Вот примеры некоторых множеств:

– множество натуральных чисел $N=\{1, 2, 3, \ldots\}$;

– множество целых чисел $Z=\{0, \pm 1, \pm 2, \pm 3, \ldots\}$;

– множество действительных чисел R;

– множество искусственных спутников, находящихся на околоземных орбитах на текущую дату;

– множество команд управления беспилотным летательным аппаратом (БЛА);

– множество зарезервированных слов языка программирования *C*++;

– множество тактико-технических характеристик БЛА;

– множество внутренних состояний подсистемы управления навигацией БЛА и т.п.

> ***Пустое множество*** – это множество, не содержащее ни одного элемента {}».

Для обозначения пустого множества используют специальный символ – $\varnothing$.

Для множеств, вне зависимости от их природы, определена специальная количественная мера – *мощность множества*, которая обладает следующими индуктивными свойствами:

– *мощность пустого множества равна* 0;

– *добавление к множеству одного элемента увеличивает его мощность на единицу;*

– *удаление из непустого множества одного элемента уменьшает его мощность на единицу.*

Введенная таким образом мера определяет, по сути, количество элементов множества. Мощность не может быть отрицательным числом. Мощность множества (или кардинальное число M) обозначается как $|M|$ либо card(M). Так, мощность множества $\{a, b, c\}$ – $|\{a, b, c\}|$ равна 3. Мощность пустого множества $|\varnothing|=0$. Понятие мощности *инвариантно* (неизменно) относительно природы элементов множества.

Равномощность. Два множества *равномощны*, если их мощности равны.

В зависимости от мощности, множества делятся на два класса (рис. 1.1): *конечные* множества ($|M|< \infty$) и *бесконечные* множества ($|M|= \infty$).

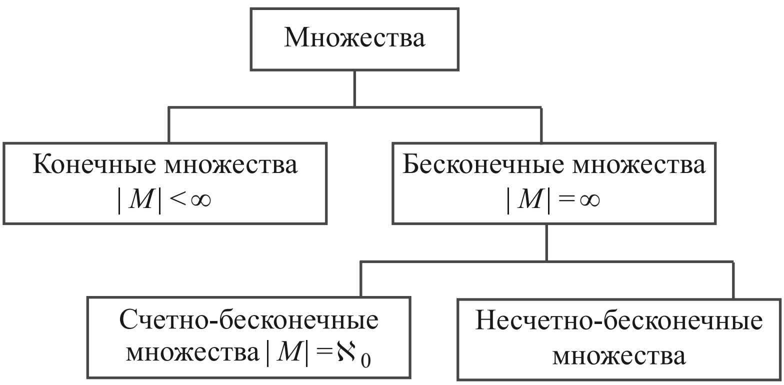


Рис. 1.1. Классификация множеств в зависимости от их мощности

Если существует взаимно однозначное соответствие между элементами бесконечного множества M и элементами множества натуральных чисел N, то множество M считается счетно-бесконечным. Из этого автоматически следует, что множество натуральных чисел – счетно-бесконечно. Для обозначения мощности счетно-бесконечных множеств используется специальный символ – $\aleph_0$ (алеф-ноль). Таким образом, мощность множества натуральных чисел $|N| = \aleph_0$.

Подмножество. Множество A является *подмножеством* множества B (обозначается как $A \subseteq B$), если из $a \in A$ следует, что $a \in B$.

Множество B в этом случае называют *надмножеством* множества A. Считается, что пустое множество $\varnothing$ является подмножеством любого множества.

Равенство множеств. Множества A и B *равны* ($A=B$), если $A \subseteq B$ и $B \subseteq A$.

Иными словами, два множества равны, если они состоят из одних и тех же элементов. Множества A и B, для которых не выполняется условие равенства, считаются неравными ($A \neq B$). Очевидно, что если $A=B$, тогда и $|A| = |B|$, однако обратное утверждение верно не всегда.

Собственное подмножество. Множество A является *собственным подмножеством* множества B (обозначается как $A \subset B$), если $A \subseteq B$ и $A \neq B$.

При этом B называют *собственным надмножеством* множества A. Пусть M – некоторое множество.

Булеан. Булеан $\mathcal{B}(M)$ множества M – это множество всех подмножеств множества M.

Булеан множества $\{m_1, m_2, m_3\}$ — $\mathcal{B}(\{m_1, m_2, m_3\})=$
$=\{\varnothing, \{m_1\}, \{m_2\}, \{m_3\}, \{m_1, m_2\}, \{m_1, m_3\}, \{m_2, m_3\}, \{m_1, m_2, m_3\}\}$.

Этот булеан содержит в качестве своих элементов восемь множеств, каждое из которых является подмножеством исходного множества.

ТЕОРЕМА. Если $|M_n| = n$, тогда $\mathcal{B}(M_n)=2^n$.

Доказательство (методом математической индукции): Пусть $T(n)$ – утверждение: «если $|M_n| = n$, то $\mathcal{B}(M_n)=2^n$», справедливость которого при любом натуральном n следует установить. Следуя методу математической индукции:

а) Если $n=1$, множество M_1 содержит один элемент, а $\mathcal{B}(M_1)$ содержит два элемента: $\varnothing$ и множество M_1, содержащее этот единственный элемент. Следовательно, $\mathcal{B}(M_1) = 2^1$ и утверждение $T(1)$ справедливо.

б) Предположим, что утверждение $T(n)$ справедливо для некоторого произвольного натурального n ($n>1$): если $|M_n| = n$, то $|\mathcal{B}(M_n)| = 2^n$.

Докажем, что в этом случае утверждение $T(n+1)$: «если $|M_{n+1}| = n+1$, то $|\mathcal{B}(M_n)| = 2^{n+1}$» также справедливо. Действительно, множество M_{n+1} можно получить из M_n введением в него нового элемента m_{n+1}. Тогда $\mathcal{B}(M_{n+1})$ будет содержать все элементы $\mathcal{B}(M_n)$ (их 2^n, как следует из б) и все элементы $\mathcal{B}(M_n)$, в каждый из которых введен элемент m_{n+1} (их также 2^n). Таким образом, $|\mathcal{B}(M_{n+1})| = 2^n + 2^n = 2^{n+1}$, что и требовалось доказать.

Универсум (универсальное множество). *Универсум (I)* – это множество, состоящее из всех элементов исследуемой области.

Универсум – это множество, в рамках которого рассматриваются любые другие множества. Универсум – это граница, за которую нельзя выйти при любых операциях на множествах. В качестве примера универсума можно рассмотреть множество студентов МАИ. Среди этого множества можно выделить подмножество студентов, которые учатся на третьем курсе, подмножество студентов, поступивших на первый курс, подмножество студентов, получающих стипендию и т.п.

Универсальное множество вне зависимости от его природы принято изображать с помощью диаграмм Венна (1834–1923) как множество точек внутри абстрактного прямоугольника с границами (рис. 1.2), в верхнем правом углу которого расположен символ универсума – *I*. Если граница диаграммы сплошная (рис. 1.2, *а*), мощность универсума конечна. Если прерывистая (рис. 1.2, *б*), мощность универсума бесконечна.

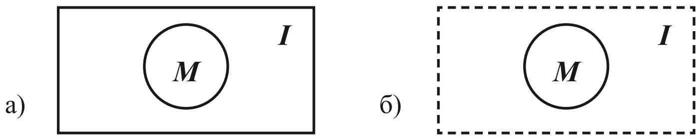


Рис. 1.2. Диаграмма Венна:
а) для конечного универсума; б) для бесконечного универсума

Любое множество в рамках универсума представляется на диаграмме кругами с соответствующим буквенным обозначением (M). Такой способ графического изображения множеств носит название кругов Эйлера (1707–1783).

На рис. 1.3 приведены примеры диаграмм (круги Эйлера) для множеств M_a и M_b.

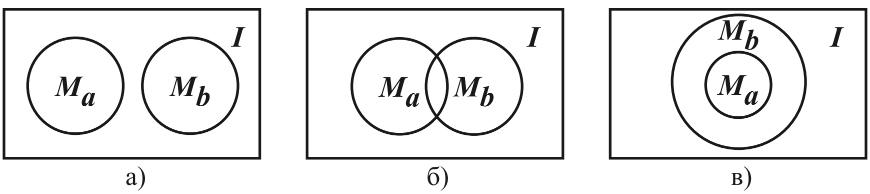


Рис. 1.3. Круги Эйлера:
а) множества M_a и M_b не имеют общих элементов;
б) множества M_a и M_b имеют общие элементы;
в) $M_a \subseteq M_b$

При исследовании множеств и подмножеств можно столкнуться с коллизиями, парадоксами, которые следуют из отсутствия конструктивного определения множества. В частности, известен так называемый парадокс Рассела (Бертран Рассел, 1872–1970).

Пусть множество M – множество всех множеств, каждое из которых не содержит себя в качестве элемента, т.е. $M=\{A|A\notin A\}$. Возникает вопрос: принадлежит ли множеству M в качестве его элемента само множество M? Возможно два варианта ответа: «да» либо «нет».

Если предположить, что ответ положительный ($M\in M$), тогда из свойства множества M ($A\notin A$) следует, что $M\notin M$.

Если предположить, что ответ отрицательный ($M\notin M$), тогда из свойства множества M ($A\notin A$) следует, что $M\in M$.

Таким образом получается, что M одновременно принадлежит и не принадлежит множеству M, что недопустимо.

1.2. Способы задания множеств

Задание множества предполагает способ однозначного определения элементов множества. Способы задания важны для описания (моделирования) различных прикладных систем и для решения задач, использующих понятие множества. Для задания множества необходимо в том или ином виде указать элементы, которые ему принадлежат. В основном используются следующие способы:

1) задание *перечислением* элементов множества;
2) задание указанием *характеристического свойства* множества;
3) задание с указанием *производящей (порождающей) функции*.

Первый способ заключается в непосредственном перечислении всех элементов множества. Способ очень прост, но применим лишь для задания конечных множеств, имеющих относительно небольшую мощность. Если число элементов множества значительно, трудоемкость его задания может превзойти все ожидаемые оценки. Ниже в качестве примера приведено множество букв латинского алфавита:

$\{a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z\}$.

Второй способ не предполагает явное перечисление элементов множества, однако характеристическое свойство множества позволяет получить ответ на вопрос о принадлежности элемента множеству – $M=\{m|P(m)\}$.

При этом способе m – обобщенное обозначение элемента множества M, а $P(m)$ – характеристическое свойство, которому удовлетворяют все элементы M. Так, множество $\{a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z\}$ можно компактно задать как

$M = \{m \mid P(m)$: «m – буква латинского алфавита»$\}$.

Данный способ применим для задания как конечных, так и бесконечных множеств. Его широко используют в математическом анализе при доказательствах теорем. Недостатком этого способа является тот факт, что для каждого элемента из I необходимо проверять (вычислять) выполнимость характеристического свойства $P(m)$, а это может быть сопряжено с высокой трудоемкостью их проведения.

Третий способ также не предполагает явное перечисление элементов множества. Способ основан на функции вычисления (порождения) некоторой (в том числе и бесконечной) последовательности элементов $m_1, m_2, \ldots, m_j, \ldots$, которые, собственно, и образуют множество. Способ включает задание некоторой функции φ от k аргументов – $\varphi(m_1, m_2, \ldots, m_k)$ и явное перечисление первых k элементов множества. Функция φ вычисляет (порождает) элемент $m_j = \varphi(m_{j-k}, \ldots, m_{j-2}, m_{j-1})$ в зависимости от значений k предыдущих элементов последовательности для каждого j $(j \geq k+1)$:

$M = \{m_1, m_2, \ldots, m_k, m_j = \varphi(m_{j-k}, \ldots, m_{j-2}, m_{j-1}) \mid j \geq k+1\}$.

При таком способе порождения в последовательностях возможно появление повторяющихся элементов. Однако множество, порожденное таким способом, не должно содержать повторяющихся элементов, и, следовательно, повторяющиеся элементы необходимо удалить. В качестве примера ниже задано множество чисел Фибоначчи – F, элемент m_j которого задается функцией φ: $m_j = \varphi(m_{j-1}, m_{j-2}) = m_{j-1} + m_{j-2}$. Для этого множества $k=2$, а начальными элементами последовательности являются $m_1=1$, $m_2=1$. Функция φ позволяет находить очередной элемент множества чисел Фибоначчи для $j \geq 2$:

$F = \{1, 2, 3, 5, 8, \ldots, m_j = m_{j-1} + m_{j-2}, \ldots \mid j \geq 2\}$.

К недостаткам этого способа следует отнести тот факт, что для ответа на вопрос о принадлежности некоторого элемента m_j множеству необходимо предварительно вычислить все предшествующие данному элементы множества.

1.3. Операции над множествами

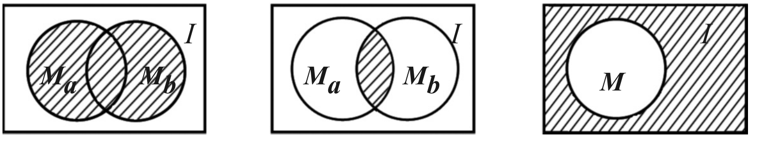
Операции над множествами позволяют «вычислять», строить новые множества из некоторой совокупности множеств подобно тому, как в арифметике вычисляются значения арифметических выражений. К основным операциям над множествами относятся операции *объединения*, *пересечения* и *дополнения*.

Объединение ($\cup$). *Объединение* двух множеств M_a и M_b есть множество, элементы которого удовлетворяют следующему условию: $M_a \cup M_b = \{x | x \in M_a$ или $x \in M_b\}$.

Пересечение ($\cap$). *Пересечение* двух множеств M_a и M_b есть множество, элементы которого удовлетворяют следующему условию: $M_a \cap M_b = \{x | x \in M_a$ и $x \in M_b\}$.

Дополнение ($^{-}$). *Дополнение* множества M (до универсума) есть множество, элементы которого удовлетворяют следующему условию: $\overline{M} = \{x | x \notin M$ и $x \in I\}$.

На рис. 1.4 приведена графическая интерпретация описанных операций на диаграммах Эйлера–Венна.



а) б) в)

Рис. 1.4. Графическая интерпретация операций:
а) объединения, б) пересечения, в) дополнения

Если использовать задание множеств с помощью характеристических свойств $M_a = \{x| P_a (x)\}$ и $M_b = \{x| P_b(x)\}$, то результаты выполнения перечисленных операций можно определить как:

$M_a \cup M_b = \{x| P(x)$: «$P_a(x)$ или $P_b(x)$»$\}$;
$M_a \cap M_b = \{x| P(x)$: «$P_a(x)$ и $P_b(x)$»$\}$;
$\overline{M}_a = \{x| \overline{P_a(x)}$: «неверно, что $P_a(x)$»$\}$.

При вычислениях множественных выражений необходимо учитывать старшинство операций. Старшей считается операция дополнения, затем пересечения, затем объединения. Для изменения порядка вычислений используют скобки. Операции, заключенные в скобки, всегда имеют высший приоритет по отношению к другим операциям. Можно считать, что операция дополнения по старшинству приравнивается к операциям, стоящим в скобках.

Операции над множествами характеризуются определенными мощностными свойствами.

ТЕОРЕМА. Мощность объединения двух множеств равна сумме их мощностей за вычетом мощности их пересечения:
$|M_a \cup M_b| = |M_a| + |M_b| - |M_a \cap M_b|$.

ТЕОРЕМА. Мощность пересечения двух множеств не превосходит мощности каждого из них:
$|M_a \cap M_b| \leqslant \min\{|M_a|, |M_b|\}$.

Следствие 1. Если $M_b \subseteq M_a$, то $|M_a \cap M_b| = |M_b|$.

Справедливо и обратное утверждение: если $|M_a \cap M_b| = |M_b|$, то $M_b \subseteq M_a$.

Следствие 2. Если $M_b \subseteq M_a$, то $|M_a \cup M_b| = |M_a|$.

Справедливо и обратное утверждение: если $|M_a \cup M_b| = |M_a|$, то $M_b \subseteq M_a$.

ТЕОРЕМА. Мощность дополнения множества M_a для конечного универсума равна разности мощности универсального множества I и мощности множества M_a: $|\overline{M_a}| = |I| - |M_a|$.

Часто в рассмотрение включаются дополнительные операции на множествах. К ним относят операцию *разности* и операцию *симметрической разности*.

Разность (\). *Разность* двух множеств M_a и M_b есть множество, элементы которого включают все те элементы M_a, которые не содержатся в M_b: $M_a \setminus M_b = \{x \mid x \in M_a \text{ и } x \notin M_b\}$.

Симметрическая разность (Δ). *Симметрическая разность* двух множеств M_a и M_b есть множество, которое состоит из всех элементов множеств M_a и M_b за исключением тех, которые содержатся как в M_a, так и в M_b:
$M_a \Delta M_b = \{\mathrm{x} \mid (x \in M_a \text{ и } x \notin M_b) \text{ или } (x \in M_b \text{ и } x \notin M_a)\}$.

Эти операции имеют низший приоритет по сравнению с операциями объединения, пересечения и дополнения. Порядок вычисления операций разности и симметрической разности регулируется только скобками.

Пусть M – некоторое множество и $S = \{M_1, M_2, \ldots, M_k\}$ – совокупность некоторых подмножеств множества $M (k > 1)$.

Разбиение множества. Совокупность k множеств $S = \{M_1, M_2, \ldots, M_k\}$ называется *разбиением* множества M на k подмножеств, если:

a) $M_1 \cup M_2 \cup \ldots \cup M_k = M$;

b) $M_i \subseteq M$ $(1 \le i \le k)$;

c) $M_i \cap M_j = \varnothing$, если $i \ne j$ $(1 \le i \le k, 1 \le j \le k)$.

Покрытие множества. Совокупность k множеств $S = \{M_1, M_2, \ldots, M_k\}$ называется *покрытием* множества M, если:

a) $M_i \subseteq M$ $(i = 1, 2, ..., k)$;

b) для каждого элемента $m \in M$ существует $M_i \in S$ $(1 \le i \le k)$, для которого $m \in M_i$;

c) удаление из S хотя бы одного множества приводит к нарушению предыдущего свойства.

Из определений следует, что любое разбиение является покрытием, но не любое покрытие является разбиением.

Пример: Пусть заданы множества $M=\{1, 2, 3, 4, 5, 6, 7, 8\}$, $M_1=$ $=\{1, 3\}$, $M_2=\{2, 4, 8\}$, $M_3=\{5, 7, 6\}$, $M_4=\{3, 4, 5, 6, 8\}$, $M_5=\{1, 2, 3, 5\}$. Тогда $S=\{M_1, M_2, M_3\}$ является и разбиением, и покрытием M. Совокупность $S=\{M_3, M_4, M_5\}$ есть покрытие, но не разбиение.

$S=\{M_1, M_3, M_4, M_5\}$ не является ни разбиением, ни покрытием.

1.4. Алгебра множеств

Результаты, которые были отражены в предыдущем разделе, определяют вычислительные возможности основных операций над множествами. Они позволяют вычислять значение множественного выражения, если известны значения входящих в него множеств (операндов), подобно вычислению арифметических выражений. Но введенные операции обладают совокупностью свойств, которые не зависят от конкретных множеств (операндов). А это, в свою очередь, позволяет определить алгебру множеств.

Пусть $M=\{M_1, M_2,\ldots, M_n\}$ – совокупность некоторых произвольных множеств, определенных на универсуме I, среди которых есть, по крайней мере, одно непустое множество. Пусть $I(M)$ – множество всех множеств, которые могут быть получены из совокупности множеств M посредством операций объединения, пересечения и дополнения.

Алгебра множеств ($А_М$) определяется как совокупность носителя – $I(M)$ и сигнатуры (операций) – $\cap$, $\cup$, $^{-}$:

$А_М=\langle I(M), \cap, \cup, ^{-}\rangle$.

Вне зависимости от множеств M_a, M_b, и M_c из $I(M)$, для операций (сигнатуры) алгебры множеств справедливы следующие свойства:

– *свойства идемпотентности:*

$M_a \cup M_a = M_a$;

$M_a \cap M_a = M_a$;

– *свойства коммутативности:*

$M_a \cup M_b = M_b \cup M_a$;

$M_a \cap M_b = M_b \cap M_a$;

– *свойства ассоциативности:*

$M_a \cup M_b \cup M_c = M_a \cup (M_b \cup M_c)$;

$M_a \cap M_b \cap M_c = M_a \cap (M_b \cap M_c)$;

– *свойства дистрибутивности:*

$M_a \cap (M_b \cup M_c) = M_a \cap M_b \cup M_a \cap M_c$;

$M_a \cup M_b \cap M_c = (M_a \cup M_b) \cap (M_a \cup M_c)$;

– *свойства де-Моргана*:

$\overline{M_a \cup M_b} = \overline{M}_a \cap \overline{M}_b$;

$\overline{M_a \cap M_b} = \overline{M}_a \cup \overline{M}_b$;

– *свойство двойного дополнения* $\overline{\overline{M}}_a = M_a$;

– *свойства универсального и пустого множеств*

$M_a \cap I = M_a$; $\quad M_a \cup I = I$;

$M_a \cap \varnothing = \varnothing$; $\quad M_a \cup \varnothing = M_a$;

$M_a \cap \overline{M}_a = \varnothing$; $\quad M_a \cup \overline{M}_a = I$.

Дополнительно к свойствам относят также свойства поглощения и склеивания, которые могут быть выведены из приведенных выше:

– *свойства поглощения:*

$M_a \cap (M_a \cup M_b) = M_a$;

$M_a \cup M_a \cap M_b = M_a$;

– *свойства склеивания:*

$M_a \cap M_b \cup M_a \cap \overline{M}_b = M_a$;

$(M_a \cup M_b) \cap (M_a \cup \overline{M}_b) = M_a$.

Для проверки справедливости перечисленных свойств можно воспользоваться диаграммами Эйлера–Венна: построить для общего случая соотношения исходных множеств диаграмму левой части проверяемого свойства и диаграмму правой части того же свойства. Если при этом структуры результирующих множеств правой и левой частей совпадают, соответствующее свойство выполняется.

На рис. 1.5 приведена графическая интерпретация второго свойства дистрибутивности $M_a \cup M_b \cap M_c = (M_a \cup M_b) \cap (M_a \cup M_c)$. Диаграммы Эйлера–Венна для левой и правой частей свойства представлены на рис. 1.5, *а* и 1.5, *б* соответственно. Из приведенных диаграмм видно, что структурно множество левой части свойства совпадает с множеством правой части.

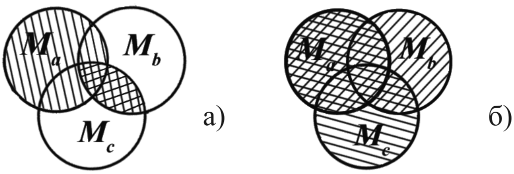


Рис. 1.5. Графическая интерпретация второго свойства дистрибутивности:
а) для левой части свойства; б) для правой части свойства

Проверить выполнение равенства можно и аналитически. Для этого необходимо показать, что левая часть есть подмножество правой, а правая – подмножество левой. Так, для проверки свойства дистрибутивности $M_a \cup M_b \cap M_c = (M_a \cup M_b) \cap (M_a \cup M_c)$ необходимо показать, что $M_a \cup M_b \cap M_c \subseteq (M_a \cup M_b) \cap (M_a \cup M_c)$ и что $(M_a \cup M_b) \cap (M_a \cup M_c) \subseteq M_a \cup M_b \cap M_c$.

Пусть $x \in M_a \cup M_b \cap M_c$. Это означает, что $x \in M_a$ или $x \in M_b \cap M_c$. Если $x \in M_a$, из этого следует, что $x \in M_a \cup M_b$ и $x \in M_a \cup M_c$, и, следовательно, $x \in (M_a \cup M_b) \cap (M_a \cup M_c)$. Если же $x \in M_b \cap M_c$, то $x \in M_b$ и $x \in M_c$ и, следовательно, $x \in M_a \cup M_b$ и $x \in M_a \cup M_c$. А из определения пересечения следует, что $x \in (M_a \cup M_b) \cap (M_a \cup M_c)$. Таким образом, показано, что $M_a \cup M_b \cap M_c \subseteq (M_a \cup M_b) \cap (M_a \cup M_c)$.

Пусть теперь $x \in (M_a \cup M_b) \cap (M_a \cup M_c)$. Это означает, что $x \in M_a \cup M_b$ и $x \in M_a \cup M_c$, что, в свою очередь, возможно, если $x \in M_a$ или $x \in M_b \cap M_c$. Если $x \in M_a$, тогда $x \in M_a \cup M_b \cap M_c$. Если $x \in M_b \cap M_c$, тогда $x \in M_a \cup M_b \cap M_c$. Таким образом, показано, что $(M_a \cup M_b) \cap (M_a \cup M_c) \subseteq M_a \cup M_b \cap M_c$. С учетом полученного выше результата можно утверждать, что $(M_a \cup M_b) \cap (M_a \cup M_c) = M_a \cup M_b \cap M_c$.

Справедливость второго свойства поглощения $M_a \cup M_a \cap M_b = M_a$ и первого свойства склеивания $M_a \cap M_b \cup M_a \cap \overline{M}_b = M_a$ можно доказать аналитически с использованием свойств алгебры множеств:

$M_a \cup M_a \cap M_b = M_a \cap I \cup M_a \cap M_b = M_a \cap (I \cup M_b) = M_a \cap I = M_a$ – свойство поглощения доказано;

$M_a \cap M_b \cup M_a \cap \overline{M}_b = M_a \cap (M_b \cap \overline{M}_b) = M_a \cap I = M_a$ – свойство склеивания доказано.

Операции объединения, пересечения и дополнения позволяют выразить и введенные ранее операции разности «\» и симметрической разности «Δ». Действительно, $M_a \setminus M_b = M_a \cap \overline{M}_b$, а

$M_a \Delta M_b = M_a \cap \overline{M}_b \cup \overline{M}_a \cap M_b$.

Понятие подмножества можно выразить с помощью операций алгебры множеств: $M_a \subseteq M_b$ эквивалентно $M_a \cap M_b = M_a$, или $M_a \cup M_b = M_b$, или $M_a \cap \overline{M}_b = \varnothing$ или, $\overline{M}_a \cup M_b = I$.

- **Задачи и упражнения**

① Для множеств $A=\{0, 1, 3, 8\}$, $B=\{0, 1, 2, 6\}$ и $C=\{0, 1, 2\}$, которые определены на универсуме $I=\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, вычислить следующие множества:

- $\overline{A \cap B} \cap (\overline{A} \cup \overline{C},)$;
- $\overline{\overline{A} \cup \overline{C} \cap B}$;

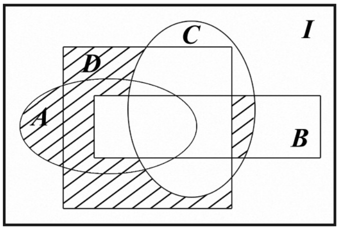
q $(A\cup\overline{B})\cap\overline{B\cap\overline{C}}$.

② Вычислить значение множества $\overline{\overline{C}\cap B\cup A}$, если известно, что $B\cap C=B$.

③ Вычислить значение множества $\overline{\overline{A}\cup B\cup\overline{C}}$, если известно, что $A\subseteq B$.

④ Вычислить значение множества $A\cup B\cap C$, если известно, что $A=B\cup C$.

⑤ Для заданной диаграммы Эйлера–Венна с определенными на ней множествами A, B, C и D задать затененное множество наиболее компактным способом.



⑥ Штат управляющей компании насчитывает 100 сотрудников, из них:

- ✓ 28 – заняты управлением производством;
- ✓ 19 – заняты управлением персоналом;
- ✓ 15 – заняты управлением финансами;
- ✓ 7 – заняты управлением производством и персоналом;
- ✓ 6 – заняты управлением производством и финансами;
- ✓ 6 – заняты управлением персоналом и финансами;
- ✓ 3 – заняты управлением производством, персоналом и финансами.

Необходимо определить:

а) сколько сотрудников занято только управлением производством;

б) сколько сотрудников не связано ни с управлением производством, ни с управлением финансами, ни с управлением персоналом.

2. ОТНОШЕНИЯ

2.1. Декартово произведение и отношение

Пусть M_1 и M_2 – некоторые непустые множества.

Декартово произведение. *Декартово произведение* $M_1 \times M_2$ множеств M_1 и M_2 есть множество упорядоченных пар вида (m_1, m_2): $M_1 \times M_2 = \{(m_1, m_2) | \ m_1 \in M_1, \ m_2 \in M_2\}$.

Из определения следует, что декартово произведение двух множеств – это, во-первых, множество, а во-вторых, множество, элементы которого обладают внутренней структурой, порядком. Порядок элементов в паре существен. В общем случае $M_1 \times M_2 \neq M_2 \times M_1$. Для конечных множеств мощность декартова произведения $|M_1 \times M_2| = |M_1| \times |M_2|$.

Если $M_1 = M_2 = M$, то декартово произведение $M \times M = M^2$ называют квадратом множества M.

***n*-арное (*n*>1) декартово произведение**. *n-арное* ($n > 1$) *декартово произведение* $M_1 \times M_2 \times \ldots \times M_n$ множеств $M_1, M_2, \ldots, M_n$ есть множество упорядоченных последовательностей длины n (кортежей, векторов), составленных из элементов множеств $M_1, M_2, \ldots, M_n$, выписанных в порядке, в котором множества образуют произведение –

$M_1 \times M_2 \times \ldots \times M_n = \{(m_1, m_2, \ldots, m_n) | \ m_i \in M_i, \ i \in \{1, 2, \ldots, n\}\}$.

Если $M_1 = M_2 = \ldots = M_n = M$, тогда n-арное декартово произведение $\underbrace{M \times M \times \ldots \times M}_{n \text{ раз}} = M^n$ называют n-ой степенью множества M.

Если M_1, M_2 ,..., M_n являются подмножествами некоторого универсума I, n-арное декартово произведение $M_1 \times M_2 \times \ldots \times M_n$ называют односортным, в противном случае – многосортным. Очевидно, что n-ая степень множества M всегда односортна.

Мощность односортных и многосортных конечных n-арных декартовых произведений вычисляется как произведение мощностей $|M_1| \times |M_2| \times \ldots \times |M_n|$ образующих их множеств.

<u>Пример</u>. Пусть множество $M=\{x_1, x_2, x_3\}$. Тогда декартово произведение (квадрат) $M \times M = M^2$ включает девять упорядоченных пар:

$M \times M = \{(x_1, x_1), (x_1, x_2), (x_1, x_3), (x_2, x_1), (x_2, \mathrm{x}_2), (\mathrm{x}_2, x_3), (x_3, x_1), (x_3, x_2), (x_3, x_3)\}$.

Декартово произведение $M \times M \times M = M^3$ содержит двадцать семь кортежей длины 3.

Пусть $M_1 \times M_2 \times \ldots \times M_n$, (M^n) – некоторое n-арное декартово произведение (степень).

> ***n-арное отношение*** $\boldsymbol{R^{(n)}}$. *n-арное отношение* $R^{(n)}$ есть произвольное непустое подмножество n-арного декартова произведения $M_1 \times M_2 \times \ldots \times M_n$, $R^{(n)} \subseteq M_1 \times M_2 \times \ldots \times M_n$.
> Для случая когда $M_1 = M_2 = \ldots = M_n = M$, $R^{(n)} \subseteq M^n$.

Из определения следует, что n-арное отношение $R^{(n)}$ есть множество, элементами которого являются кортежи длины n.

Для случая $n=2$ отношение $R^{(2)}$ называют бинарным и обычно верхний индекс «2» опускают. Таким образом, бинарное отношение R на $M_1 \times M_2$ (на M^2):

$R \subseteq M_1 \times M_2$ есть $R=\{(m_i, m_j) | m_i \in M_1$ и $m_j \in M_2\}$,

или $R \subseteq M^2$ есть $R=\{(m_i, m_j) | m_i \in M$ и $m_j \in M\}$.

Замечание. Различие между определениями бинарных отношений $R \subseteq M_1 \times M_2$ и $R \subseteq M^2$ не имеет существенного значения. Действительно, если отношение R определено как $R \subseteq M_1 \times M_2$, то введение множества $M = M_1 \cup M_2$ позволяет представить отношение $R \subseteq M_1 \times M_2$ как $R \subseteq M^2$. И обратно: если отношение R определено как $R \subseteq M^2$, то введение множеств $M_1 = M$ и $M_2 = M$ позволяет представить отношение $R \subseteq M^2$ как $R \subseteq M_1 \times M_2$.

Если пара $(m_i, m_j) \in R$, говорят, что элемент m_i и элемент m_j находятся в отношении R. Часто вместо записи $(m_i, m_j) \in R$ (префиксная запись) применяют запись вида $m_i R m_j$ (инфиксная запись), которая читается как «элемент m_i находится в отношении R с элементом m_j».

Пусть X и Y – некоторые непустые множества и $X \times Y$ – декартово произведение.

Функция (унарная). Бинарное отношение $F \subseteq X \times Y$ есть функция (унарная), если $F = \{(x, y) | x \in X, y \in Y$ и из $(x, y_1) \in F$, $(x, y_2) \in F$ следует $y_1 = y_2\}$.

Другими словами, функция – это такое бинарное отношение, при котором каждому элементу x из X соответствует не более одного y из Y. Для функций обычно вместо записи $(x, y) \in F$ или записи xFy используют запись вида $y = F(x)$. При таком способе определения функцию удобно представлять в виде таблицы.

В левом столбце указываются значения аргумента, а в правом – соответствующее значение функции. Такой способ задания функций применяется только для случая конечных множеств.

X	Y

Примером может служить функция $F \subseteq X \times Y$, которая определена на множестве X (множество искусственных спутников Земли) и множестве Y (множество стран – владельцев этих спутников). Эта функция позволяет «вычислять», находить для любого спутника из X страну-владельца спутника из Y.

Функция (n-арная). Отношение $F \subseteq X^n \times Y$ называется n-арной функцией, если $F=\{(x_1, x_2, ..., x_n, y) | (x_1, x_2, ..., x_n) \in X^n$, $y \in Y$, $((x_1, x_2, ..., x_n, y_1) \in F$ и $(x_1, x_2, ..., x_n, y_2) \in F \rightarrow y_1=y_2)\}$.

Функция считается *полностью определенной*, если ее значение определено для каждого значения аргумента x из X. И функция считается *частично определенной*, если ее значение определено не для каждого $x \in X$.

Если $X=Y$, функцию $F \subseteq X \times X$ ($F \subseteq X^n \times X$) называют операцией, соответственно, унарной или n-арной.

2.2. Бинарные отношения

2.2.1. Способы задания бинарных отношений

Бинарные отношения имеют большое значение для теории и практики, так как все мыслимые объекты находятся между собой в определенных отношениях. Бинарные отношения – это отношения, посредством которых могут быть описаны, промоделированы отношения любой степени сложности. По сути, бинарные отношения описывают связи, структуры взаимодействия между некоторыми объектами. Хотя природа объектов может быть разнообразной, отношения между ними могут обладать определенной общностью. Для анализа этой общности вводится понятие «*равенства*» отношений.

Исходя из определения бинарного отношения как множества, два отношения R_1 и R_2 равны, если $R_1 \subseteq R_2$ и $R_2 \subseteq R_1$. Такое определение «*равенства*» является весьма сильным.

Часто исследователей интересуют не столько сами элементы множества M, на котором определено отношение R, сколько структура связей (отношение) между ними. С этой целью на множестве бинарных отношений определяется отношение структурного подобия, которое называется отношением *изоморфизма*.

Изоморфизм отношений. Два отношения $R_a \subseteq M_a^2$ и $R_b \subseteq M_b^2$ изоморфны, если существует взаимно однозначная функция φ: $M_a \rightarrow M_b$ (функция изоморфизма), такая, что:
если $m_{a1} R_a m_{a2}$, то $\varphi(m_{a1})\ R_b\ \varphi(m_{a2})$, и обратно:
если $m_{b1} R_b m_{b2}$, то $\varphi^{-1}(m_{b1})\ R_a\ \varphi^{-1}(m_{b2})$,
где φ^{-1} – функция, обратная для φ (φ^{-1}: $M_b \rightarrow M_a$).

Можно сказать, что два отношения изоморфны, если одно отношение получено из другого путем переименования (перекодирования) его элементов.

Задание бинарных отношений предполагает фиксацию элементов отношения, достаточную для проведения исследований. Способы задания важны для описания и моделирования различных прикладных, программных и информационных систем, использующих понятие бинарного отношения. Для задания бинарного отношения необходимо тем или иным способом указать элементы (упорядоченные пары), которые ему принадлежат. Иногда бывает достаточно задать отношения с точностью до изоморфизма, т.е. без предметной привязки к конкретным множествам. В основном используются следующие способы:

- задание *перечислением* элементов бинарного отношения;
- задание с указанием *характеристического свойства* бинарного отношения;
- задание с помощью *матрицы смежности*;
- *графическое* задание бинарного отношения (с помощью графа);
- *функциональный* способ задания бинарного отношения.

Способ перечисления элементов отношения заключается в непосредственном перечислении всех элементов отношения как элементов множества. Способ очень прост, но применим лишь для задания конечных отношений, имеющих относительно небольшую мощность.

В качестве примера приведено некоторое бинарное отношение $R \subseteq M^2$, определенное на множестве $M = \{m_1, m_2, m_3\}$:

$R = \{(m_1, m_1), (m_1, m_2), (m_2, m_3), (m_3, m_1), (m_3, m_2)\}$.

На практике этот способ часто реализуется в виде таблицы.

Первый элемент пары	Второй элемент пары
m_1	m_1
m_1	m_2
m_2	m_3
m_3	m_1
m_3	m_2

Такой способ *представления* широко применяется в информационных системах при построении реляционных баз данных.

При задании *отношения с указанием характеристического свойства* элементы отношения явно не перечисляются, но характеристическое свойство дает ответ на вопрос о принадлежности той или иной пары декартова произведения отношению:

$R = \{(m_i, m_j) | (m_i \in M), (m_j \in M), P(m_i, m_j)\}$.

При этом способе (m_i, m_j) – обобщенное обозначение элемента отношения R, а $P(m_i, m_j)$ – характеристическое свойство, которому удовлетворяют все элементы (упорядоченные пары) из R.

Примером такого задания является отношение $Q \subseteq M^2$, где M – множество людей, а $Q = \{(m_i, m_j) | (m_i \in M), (m_j \in M), P(m_i, m_j)$: «$m_i$ является потомком m_j»$\}$. Такое отношение лежит в основе генеалогических исследований рода человеческого.

Задание бинарного отношения с помощью *матрицы смежности* позволяет определить отношение с точностью до изоморфизма. *Матрица смежности* $M_{\text{см}}$ отношения $R \subseteq M^2$ – $M_{\text{см}}(R)$ представляет собой квадратную матрицу $|M| \times |M|$ элементов t_{ij}:

$$M_{\text{см}}(R) = \| t_{ij} \|, \text{ где } t_{ij} = \begin{cases} 1, \text{ если } (m_i, m_j) \in R, \\ 0, \text{ если } (m_i, m_j) \notin R, \end{cases}$$

и $1 \leq i \leq |M|$, $1 \leq j \leq |M|$.

Матрица смежности для отношения $R = \{(m_1, m_1), (m_1, m_2), (m_2, m_3), (m_3, m_1), (m_3, m_2)\}$ представляется как:

$$M_{\text{см}}(R) = \begin{array}{|c|c|c|}\hline 1 & 1 & 0 \\ \hline 0 & 0 & 1 \\ \hline 1 & 1 & 0 \\ \hline \end{array}$$

При *графическом* задании бинарного отношения каждому элементу $m_i \in M$ ставится в соответствие вершина, а каждой паре $(m_i, m_j) \in R$ ставится в соответствие ориентированная дуга, связь, которая соединяет вершину m_i с вершиной m_j:

вершина m_i – m_i дуга (m_i, m_j) – $m_i \longrightarrow m_j$

Графическая фигура $G(R) = \langle M, R \rangle$, полученная таким образом, называется графом (ориентированным графом) отношения R. Графическое представление отношения $R = \{(m_1, m_1), (m_1, m_2), (m_2, m_3), (m_3, m_1), (m_3, m_2)\}$, приведено на рис. 2.1.

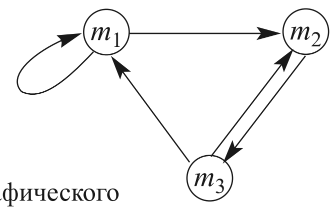


Рис. 2.1. Пример графического задания отношения R

Функциональный способ задания бинарного отношения основан на определении функции Γ отображения множества M в булеан $\mathcal{B}(M)$: $\Gamma(m) = \{m_i | (m\,R\,m_i)\}$. Иногда эту функцию называют функцией окрестности элемента (вершины) m.

Бинарное отношение $R = \{(m_1, m_1), (m_1, m_2), (m_2, m_3), (m_3, m_1), (m_3, m_2)\}$ функционально определяется как $\Gamma(m_1) = \{m_1, m_2\}$, $\Gamma(m_2) = \{m_3\}$, $\Gamma(m_3) = \{m_1, m_2\}$.

Табличное представление функции $\Gamma(m)$:

m	$\Gamma(m)$
m_1	$\{m_1, m_2\}$
m_2	$\{m_3\}$
m_3	$\{m_1, m_2\}$

Замечание. Возможен и другой способ функционального задания отношения с помощью обратной функции $\Gamma^{-1}(m)=\{m|(m_i, m)\in R\}$. Иногда такую функцию называют функцией ко-окрестности элемента (вершины) m.

2.2.2. Операции над бинарными отношениями

Для бинарных отношений $R\subseteq M^2$ как для математических объектов определено множество операций. Некоторые из них основаны на том, что отношение – это множество. К таким операциям относятся операции объединения, пересечения и дополнения. Другие операции связаны с особенностями элементов отношения – упорядоченностью пар. К таким операциям относят операции [14] инвертирования и композиции отношений.

Пусть $R_1\subseteq M^2$ и $R_2\subseteq M^2$ – некоторые отношения.

Объединение отношений $R_1\cup R_2$ есть отношение, состоящее из всех пар, входящих в R_1 или в R_2:

$R_1\cup R_2=\{(m_i, m_j)|(m_i, m_j)\in R_1$ или $(m_i, m_j)\in R_2\}$.

Пересечение отношений $R_1\cap R_2$ есть отношение, состоящее из всех пар, входящих в R_1 и в R_2:

$R_1\cap R_2=\{(m_i, m_j)|(m_i, m_j)\in R_1$ и $(m_i, m_j)\in R_2\}$.

Дополнение отношения R (до универсума) есть отношение $\overline{R}$, состоящее из всех тех пар декартова произведения $M\times M$, которые не входят в R:

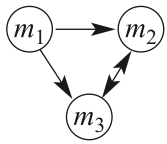
$\overline{R}=\{(m_i, m_j)|(m_i, m_j)\notin R$ и $(m_i, m_j)\in M\times M\}$.

Инвертирование отношения R есть отношение R^{-1}, состоящее из всех пар R, в которых изменен порядок следования элементов на противоположный: $R^{-1}=\{(m_i, m_j)|(m_j, m_i)\in R\}$. R^{-1} называют обратным отношением R.

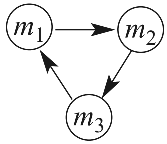
Композиция $R_1{\circ}R_2$ отношений R_1 и R_2 есть отношение, элементы которого удовлетворяют условию $R_1{\circ}R_2=\{(m_i,\ m_k)|\ (m_i,\ m_j)\in R_1$ и $(m_j,\ m_k)\in R_2\}$. Произведение вида $R{\circ}R^{-1}$ называют ядром отношения R.

Пример. Пусть на множестве $M=\{m_1,\ m_2,\ m_3\}$ заданы два бинарных отношения R_1 и R_2.

$R_1=\{(m_1, m_2), (m_1, m_3), (m_2, m_3), (m_3, m_2)\}$;

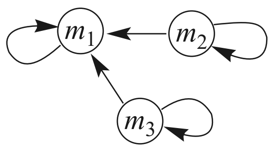


$R_2=\{(m_1, m_2), (m_2, m_3), (m_3, m_1)\}$.

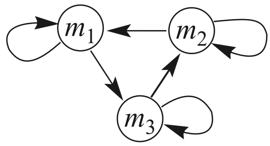


Тогда результаты выполнения операций дополнения, инвертирования, композиции и операции вычисления ядра над этими отношениями в множественной и графической формах будут иметь следующий вид:

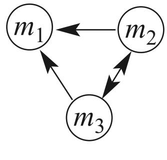
$\overline{R}_1=\{(m_1,\ m_1),\ (m_2,\ m_1),\ (m_2,\ m_2),\ (m_3,\ m_1),\ (m_3, m_3)\}$;



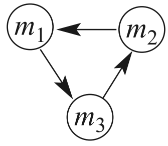
$\overline{R}_2=\{(m_1,\ m_1),\ (m_1,\ m_3),\ (m_2,\ m_1),\ (m_2,\ m_2),\ (m_3, m_2), (m_3, m_3)\}$;



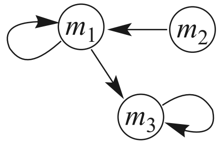
$R_1^{-1}=\{(m_2, m_1),(m_2, m_3),(m_3, m_1),(m_3, m_2)\}$;



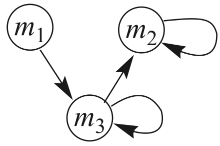
$R_2^{-1}=\{(m_1, m_3), (m_2, m_1), (m_3, m_1)\};$



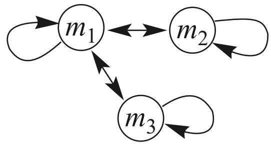
$R_1 \circ R_2=\{(m_1, m_1), (m_1, m_3), (m_2, m_1), (m_3, m_3)\};$



$R_2 \circ R_1=\{(m_1, m_3), (m_2, m_2), (m_3, m_2), (m_3, m_3)\};$



$R_1 \circ R_1^{-1}=\{(m_1, m_1), (m_1, m_2), (m_1, m_3), (m_2, m_1),$
$(m_2, m_2), (m_2, m_1), (m_3, m_3)\}.$

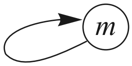


Для отношений, как и для множеств, определены понятия включения «$\subseteq$» и равенства «=». Если $R_1 \subseteq R_2$, то отношение R_1 называется *сжатием* отношения R_2, а отношение R_2 – *расширением* отношения R_1. Для сжатия справедлива следующая лемма.

ЛЕММА. Если $F \subseteq X \times Y$ – функция и F_1 есть некоторое сжатие F, то F_1 также функция.

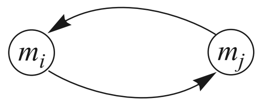
2.2.3. Свойства бинарных отношений

Пусть R – некоторое отношение на M.

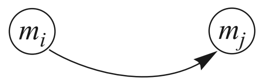
Бинарное отношение R называется *рефлексивным*, если для каждого $m \in M$ пара $(m, m) \in R$: $m \in M \rightarrow (m, m) \in R$. Свойство рефлексивности отношения диагностируется по матрице смежности наличием единичной главной диагонали. При графическом задании рефлексивного отношения R каждая вершина имеет так называемую рефлексивную дугу (петлю): 

Бинарное отношение R называется *иррефлексивным*, если для каждого элемента $m\in M$ пара $(m, m)\notin$ R: $m\in M\to(m, m)\notin R$. Матрица смежности иррефлексивного отношения содержит нулевую главную диагональ. При графическом задании иррефлексивное отношение R не содержит ни одной вершины с рефлексивной дугой.

Бинарное отношение R называется *симметричным*, если для каждой пары $(m_i, m_j)\in R$ пара (m_j, m_i) также принадлежит отношению R: $(m_i, m_j)\in R\to(m_j, m_i)\in R$. Матрица смежности симметричного отношения симметрична относительно главной диагонали. При графическом задании симметричного отношения каждая дуга $(m_i, m_j)\in R$ дополнена симметричной ей дугой (m_j, m_i).



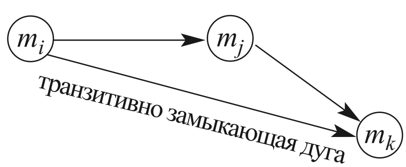
Бинарное отношение R называется *антисимметричным*, если из $(m_i, m_j)\in R$ и $(m_j, m_i)\in R$ следует, что $m_j{=}m_i$: $(m_i, m_j)\in R$ и $(m_j, m_i)\in R\to$ $\to(m_j{=}m_i)$. Матрица смежности антисимметричного отношения характеризуется определенной асимметрией: если $t_{ij}=1$, то $t_{ji}=0$. При графическом задании такого отношения нет ни одной дуги $(m_i, m_j)\in R$ (при $m_i\neq m_j$), для которой есть симметричная ей дуга (m_j, m_i).



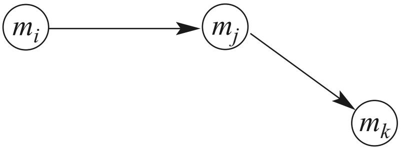
Бинарное отношение R называется *транзитивным*, если для любых трех элементов m_i, m_j, m_k из M, таких, что пара $(m_i, m_j)\in R$ и пара $(m_j, m_k)\in R$, пара (m_i, m_k) также принадлежит отношению R:

$(m_i, m_j)\in R$ и $(m_j, m_k)\in R\to(m_i, m_k)\in R$.

При графическом задании транзитивного отношения каждая пара дуг (m_i, m_j) и (m_j, m_k) дополнена так называемой *транзитивно замыкающей дугой* (m_i, m_k).



Бинарное отношение R называется *интранзитивным*, если для любых трех элементов m_i, m_j, m_k из M, таких, что пара $(m_i, m_j) \in R$, пара $(m_j, m_k) \in R$ и пара $(m_i, m_k) \in R$, следует, что $m_i = m_j$ или $m_j = m_k$ или $m_i = m_k$: $((m_i, m_j) \in R$ и $(m_j, m_k) \in R$ и $(m_i, m_k) \in R) \to (m_i = m_j)$ или $(m_j = m_k)$ или $(m_i = m_k)$. При графическом задании интранзитивного отношения для каждой пары «транзитных» дуг (m_i, m_j) и (m_j, m_k) отсутствует транзитивно замыкающая дуга (m_i, m_k).



Пусть R – некоторое бинарное отношение на M, и пусть P – некоторое свойство бинарных отношений. Бинарное отношение $\Re$ на M называют замыканием бинарного отношения R относительно свойства P – $\Re\,(R/P)$, если:

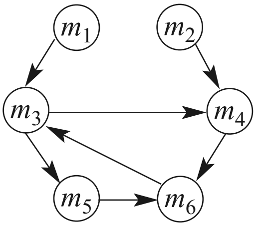
1) $R \subseteq \Re\,(R/P)$, т.е. $\Re\,(R/P)$ является расширением R;

2) свойство P выполняется для $\Re\,(R/P)$;

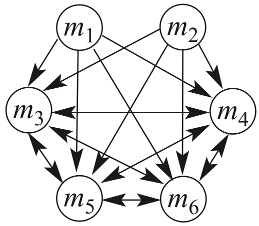
3) для любого расширения Q отношения R на M, такого, что $R \subseteq Q \subset \Re\,(R/P)$, свойство P не выполняется.

В связи с рассмотренными выше свойствами выделяют рефлексивное, симметричное и транзитивное замыкания. Замыкание отношения R по некоторому свойству P получается из исходного добавлением в него минимального числа элементов (m_i, m_j), при котором обеспечивается выполнение заданного свойства. На рис. 2.2 приведен пример некоторого бинарного отношения R и его транзитивного замыкания.

Подобным образом выполняются рефлексивное и симметричное замыкания.



а) отношение R



б) транзитивное замыкание отношения R

Рис. 2.2. Пример отношения и его транзитивного замыкания

2.3. Классы бинарных отношений

2.3.1. Отношение эквивалентности

Бинарное отношение называется отношением *эквивалентности*, если оно рефлексивно, симметрично и транзитивно.

Для обозначения отношения эквивалентности применяют символ «~». Иногда для этой цели используют символы «≡» или «↔». Для отношения эквивалентности вместо записи $(m_i, m_j) \in R$ или $m_i R m_j$ применяют запись $m_i \sim m_j$ либо $m_i \equiv m_j$. Это означает, что m_i и m_j эквивалентны в заданном отношении.

Пусть M – некоторое множество с определенным на нем отношением эквивалентности. Множество $k(m)=\{m_i | m_i \in M \text{ и } m \sim m_i\}$ – множество элементов, эквивалентных элементу m, называют классом эквивалентности, порожденным элементом m. Для отношения эквивалентности справедлива следующая теорема.

<u>ТЕОРЕМА</u>. Пусть $\langle M, \sim \rangle$ – некоторое множество с определенным на нем отношением эквивалентности, тогда:

1) если $m \in M$, то $m \in k(m)$;
2) если $m_i \in M$, $m_j \in M$, и $m_i \sim m_j$, то $k(m_i)=k(m_j)$.

Действительно, из $m \in M$ и из свойства рефлексивности отношения эквивалентности следует, что $m \sim m$ и, следовательно, $m \in k(m)$.

Для доказательства второго утверждения необходимо показать, что если $m_i \sim m_j$, то $k(m_i) \subseteq k(m_j)$ и $k(m_j) \subseteq k(m_i)$.

Пусть m_s – некоторый элемент из $k(m_i)$, $m_s \in k(m_i)$. Из определения $k(m_i)$ следует, что $m_s \sim m_i$. А из условия $m_i \sim m_j$ и из свойства транзитивности отношения эквивалентности следует, что $m_s \sim m_j$. Таким образом, $m_s \in k(m_j)$ и показано, что $k(m_i) \subseteq k(m_j)$.

Если предположить, что $m_s \in k(m_j)$, то из $m_s \sim m_j$, $m_j \sim m_i$ и из свойства транзитивности отношения эквивалентности следует, что $m_s \sim m_i$. Таким образом, $m_s \in k(m_i)$ и показано, что $k(m_j) \subseteq k(m_i)$.

Из $k(m_i) \subseteq k(m_j)$ и из $k(m_j) \subseteq k(m_i)$ следует, что $k(m_i) = k(m_j)$, что и требовалось доказать. Это утверждение показывает, что класс эквивалентности порождается любым элементом этого класса.

Прямым следствием теоремы является утверждение: если существуют два класса эквивалентности k_1 и k_2, включающие общий элемент $m \in M$, то $k_1 = k_2$.

<u>ТЕОРЕМА</u>. Отношение эквивалентности $\langle M, \sim \rangle$ определяет на M *разбиение* на классы эквивалентных между собой элементов.

Действительно, пусть $S = \{k_1, k_2, \ldots, k_p\}$ – совокупность классов эквивалентности на M. Из определения класса эквивалентности $k_i \subseteq M$. Из предыдущего утверждения следует, что $k_i \cap k_j = \varnothing$, если $i \neq j$. Учитывая, что каждый элемент из $m \in M$ порождает класс эквивалентности, справедливо, что $k_1 \cup k_2 \cup \ldots \cup k_p = M$.

Отношение эквивалентности имеет большое значение для практики и теории, так как позволяет выполнять обобщение, проводить «*сжатие*» информации: свойства и характеристики *любого* представителя класса эквивалентности выполняются и для всех остальных элементов этого класса.

<u>Пример отношения эквивалентности</u>. Пусть отношение R определено на множестве натуральных чисел N как: $R = \{(m_i, m_j) | m_i \in N, m_j \in N, m_i \bmod 3 = m_j \bmod 3\}$. Иными словами, пара натуральных чисел

принадлежит отношению, если совпадают их остатки при целочисленном делении на три. Это отношение:

- рефлексивно, так как для любого $m_i \in N \to m_i$ mod 3= m_i mod 3, и, следовательно, $(m_i, m_i) \in R$;
- симметрично, так как если $(m_i, m_j) \in R$, тогда m_i mod 3= m_j mod 3, и, следовательно, в силу свойства коммутативности функции mod вычисления остатка, m_j mod 3= m_i mod 3 $\to (m_j, m_i) \in R$;
- транзитивно, что также следует из свойства функции mod.

Следовательно, заданное отношение является отношением эквивалентности.

Если учесть, что при делении на три возможно получение одного из трех остатков – 0, 1 или 2, это отношение определяет три класса эквивалентности:

класс $k_0 = \{m_i | m_i \in N \text{ и } m_i \bmod 3=0\}$;

класс $k_1 = \{m_i | m_i \in N \text{ и } m_i \bmod 3=1\}$;

класс $k_2 = \{m_i | m_i \in N \text{ и } m_i \bmod 3=2\}$.

2.3.2. Отношение порядка

Бинарное отношение называется *отношением порядка*, если оно:

- рефлексивно;
- антисимметрично;
- транзитивно.

Отношение порядка обычно обозначается символом «$\leq$». Для такого отношения вместо записи $(m_i, m_j) \in R$ или $m_i R m_j$ используется запись $m_i \leq m_j$. Множество M с определенным на нем отношением порядка $\langle M, \leq \rangle$ считается упорядоченным этим отношением.

Бинарное отношение называется *отношением строгого порядка*, если оно:

- иррефлексивно;
- антисимметрично;
- транзитивно.

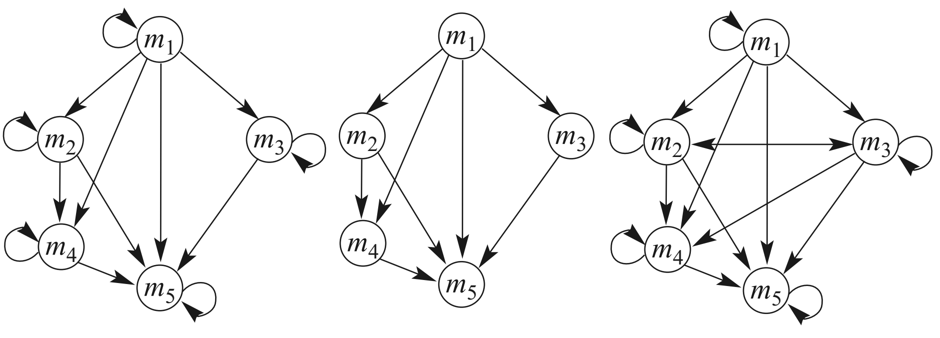
Отношение строгого порядка обычно обозначают символом «$<$», а множество M с определенным на нем отношением строгого порядка – как $\langle M, <\rangle$.

Бинарное отношение называется *отношением предпорядка*, если оно:

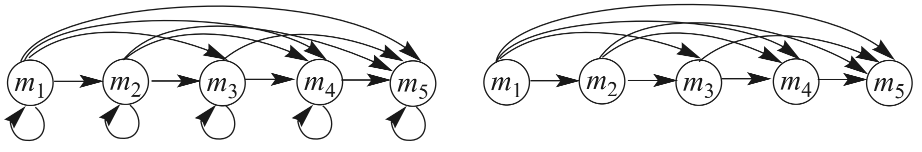
- рефлексивно;
- несимметрично;
- транзитивно.

Два элемента m_i и m_j упорядоченного множества M сравнимы, если $m_i \leq m_j$ либо $m_j \leq m_i$. Если любые два элемента m_i и m_j упорядоченного множества M сравнимы, то отношение порядка называют отношением *линейного* порядка. Если это не выполняется, то отношение порядка считается отношением *частичного* порядка.

На рис. 2.3 приведены примеры введенных понятий: частичного порядка, строгого частичного порядка, предпорядка, линейного



а) частичный порядок б) строгий частичный порядок в) предпорядок



г) линейный порядок д) линейный строгий порядок

Рис. 2.3. Примеры отношений порядка

порядка и линейного строгого порядка, заданных графически. Каждая дуга (m_j, m_i) здесь понимается как $m_j \leq m_i$. $m_i \longrightarrow m_j$

Для отношения порядка справедлива следующая теорема.

ТЕОРЕМА. Любое подмножество упорядоченного множества упорядочено.

Из этой теоремы следует, что подмножество линейно упорядоченного множества также линейно упорядочено.

Далее по тексту, если это не оговаривается специально, будут рассматриваться только отношения порядка. Пусть m_i и m_j – два элемента упорядоченного множества M. Говорят, что элемент m_j *покрывает* элемент m_i, (обозначение – $m_i \prec m_j$) если:

– $m_i \leq m_j$;

– $m_i \neq m_j$;

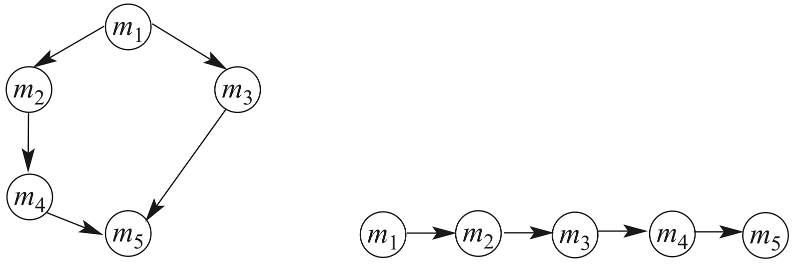
– не существует такого элемента m в M, отличного от m_i и m_j, для которого справедливо $m_i \leq m \leq m_j$.

Введенное отношение покрытия «$\prec$» для упорядоченных множеств позволяет строить более компактное графическое представление упорядоченных множеств в виде *диаграммы Хассе*. Действительно, из определения отношения покрытия следует, что оно иррефлексивно, антисимметрично и интранзитивно. При его графическом представлении (в диаграмме Хассе) будут отсутствовать все рефлексивные и транзитивные дуги исходного отношения порядка. Рис. 2.4 демонстрирует примеры диаграмм Хассе для отношений частичного и линейного порядка, которые были приведены выше (рис. 2.3, *а, б, г, д*).

Таким образом, диаграмма Хассе задает бинарное отношение покрытия, которое обладает следующими свойствами:

– его транзитивное и рефлексивное *замыкание* эквивалентно исходному отношению порядка $\langle M, \leq \rangle$;

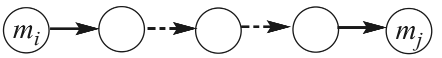
– любое его *сужение* приводит к нарушению предыдущего свойства.



а) частичный порядок б) линейный порядок

Рис. 2.4. Примеры диаграмм Хассе для отношения порядка

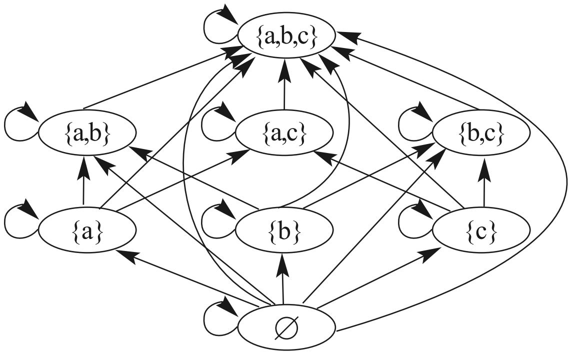
Возникает вопрос: можно ли непосредственно по диаграмме Хассе (по отношению покрытия) восстановить отношение «≤» на M без построения транзитивного и рефлексивного замыкания? Да, это возможно. Если диаграмма Хассе содержит последовательность линейно упорядоченных вершин (цепь), которая начинается с m_i и за-



канчивается на m_j, это означает, что $m_i \le m_j$. Иными словами, два элемента множества M сравнимы в отношении порядка, если в соответствующей диаграмме Хассе существует цепь, в которой эти элементы являются ее начальной и конечной вершинами. Число дуг в цепи называют длиной цепи.

Замечание. Без нарушения общности полагается, что в диаграмме Хассе допустимы цепи длины 0, а это означает, что каждый элемент сравним сам с собой – $m_i \le m_i$.

На рис. 2.5 приведен пример графического представления и соответствующей диаграммы Хассе для отношения включения: $m_i \in M$, $M = \mathcal{B}(\{a, b, c\})$ и $m_i \le m_j$, если $m_i \subseteq m_j$.



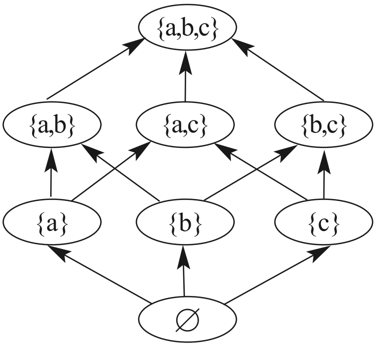


Рис. 2.5. Пример графического представления отношения порядка и соответствующей диаграммы Хассе

2.3.3. Структурные свойства упорядоченных множеств

Пусть M – некоторое множество с заданным на нем отношением порядка «$\leq$» – $\langle M,\leq\rangle$.

Элемент m_α, $m_\alpha \in M$ называется *максимальным* элементом множества M, если для любого элемента m из M условие $m_\alpha \leq m$ справедливо только тогда, когда $m_\alpha = m$. Это означает, что среди элементов M не существует таких m, что $m_\alpha < m$.

Элемент m_β, $m_\beta \in M$ называется *минимальным* элементом множества M, если для любого элемента m из M условие $m \leq m_\beta$ справедливо только тогда, когда $m_\beta = m$. Это означает, что среди элементов множества M не существует такого m, что $m < m_\beta$.

<u>ТЕОРЕМА</u>. Любое непустое конечное упорядоченное множество имеет хотя бы один максимальный и хотя бы один минимальный элемент.

Элемент m_α, $m_\alpha \in M$ называется *наибольшим* элементом множества M, если отношение $m \leq m_\alpha$ выполняется для любого элемента m из M.

ТЕОРЕМА. Любое упорядоченное множество имеет не более одного наибольшего (наименьшего) элемента.

Доказательство (для случая наибольшего). Предположим обратное: в упорядоченном множестве M существует два (т.е. более одного) неравных между собой наибольших элемента $m_{\alpha1}$ и $m_{\alpha2}$. Из определения того, что $m_{\alpha1}$ – наибольший элемент в M, следует, что $m_{\alpha2} \leq m_{\alpha1}$. А из того, что $m_{\alpha2}$ – наибольший элемент в M, следует, что $m_{\alpha1} \leq m_{\alpha2}$. В силу свойства антисимметричности отношения порядка, из ($m_{\alpha2} \leq m_{\alpha1}$) и из ($m_{\alpha1} \leq m_{\alpha2}$) следует $m_{\alpha1} = m_{\alpha2}$. Таким образом, два наибольших элемента в упорядоченном множестве возможны только тогда, когда они равны. А это противоречит предположению о том, что упорядоченное множество может иметь два неравных между собой наибольших элемента.

Пусть M' – некоторое подмножество упорядоченного множества M. Учитывая, что подмножество упорядоченного множества также упорядочено, для M' определены понятия максимального, минимального, наибольшего и наименьшего элементов. Элемент m_α, $m_\alpha \in M$ называется *верхней границей* (*мажорантой*) множества M', если для любого элемента m из M', $m \leq m_\alpha$. Соответственно, элемент m_β, $m_\beta \in M$ называется *нижней границей* (*минорантой*) множества M', если для любого элемента m из M', $m_\beta \leq m$.

Наименьшая верхняя граница множества M' называется *супремумом* (*верхней гранью*) множества M' – $\sup(M')$. Наибольшая нижняя грань множества M' называется *инфинумом* (*нижней гранью*) множества M' – $\inf(M')$. Из введенных определений следует, что любое множество M' не может иметь более одного супремума и более одного инфинума. Более того, если $\sup(M')$ принадлежит множеству M', то $\sup(M')$ совпадает с наибольшим элементом M'. И если $\inf(M')$ принадлежит множеству M', то $\inf(M')$ совпадает с наименьшим элементом M'.

Если множество верхних граней упорядоченного множества M имеет наибольший элемент, такой элемент называют *универсальной* *верхней гранью* и обозначают символом *I*. Соответственно, если множество нижних граней упорядоченного множества M имеет наименьший элемент (*наименьшая верхняя грань*), такой элемент называют *универсальной нижней гранью* и обозначают символом *0*.

Для упорядоченных множеств справедлив *принцип двойственности*: отношение, обратное отношению порядка, также является отношением порядка. Отношение, обратное для отношения порядка, называют *двойственным* отношением. Принцип двойственности позволяет принимать без доказательства для двойственного отношения все утверждения, справедливые для исходного отношения порядка.

Действительно обращение отношения порядка не нарушает свойств рефлексивности, антисимметричности и транзитивности. При графическом задании обращение отношения сводится к изменению ориентации стрелок на противоположное.

Принцип *двойственности* позволяет все утверждения, справедливые для «$\le$», распространить и на «$\le^{-1}$».

Из определений экстремальных характеристик упорядоченных множеств можно получить следующие утверждения:

- в любом подмножестве M' упорядоченного множества M существует не менее одного максимального и не менее одного минимального элемента;
- если в подмножестве M' упорядоченного множества M существует наибольший элемент, то он совпадает с $\sup(M')$;
- если в подмножестве M' упорядоченного множества M существует наименьший элемент, то он совпадает с $\inf(M')$;
- если в подмножестве M' упорядоченного множества M существует более одного максимального элемента, то не существует наибольшего элемента M';
- если в подмножестве M' упорядоченного множества M существует более одного минимального элемента, то не существует наименьшего элемента M';

- если множество максимальных и множество минимальных элементов подмножества M' упорядоченного множества M совпадают, то M' состоит из попарно не сравнимых элементов.

Пусть, как и выше, M – некоторое упорядоченное множество. Если для любых двух элементов m_i и m_j из M существует $\sup(\{m_i,m_j\})$ и $\inf(\{m_i,m_j\})$, то такое упорядоченное множество называют решеткой. Понятие решетки широко используется при анализе различных структур, а также для исследования свойств различных алгебраических систем.

2.3.4. Линеаризация отношения частичного порядка

Пусть на множестве M заданы два отношения порядка:

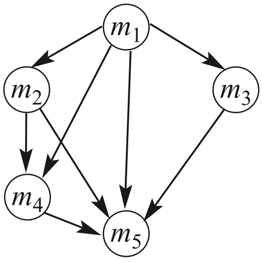
$R_1 \subset M \times M$ – отношение частичного порядка;

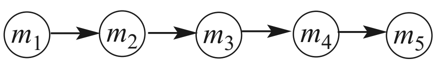
$R_2 \subset M \times M$ – отношение линейного порядка.

Говорят, что отношение R_2 является линеаризацией отношения R_1, если $R_1 \subseteq R_2$. Таким образом, линеаризация означает расширение отношения до линейного порядка с помощью введения дополнительных элементов пар. При этом не нарушается отношение сравнимости для элементов исходного отношения R_1. Всегда ли линеаризация возможна? Ответ на этот вопрос содержится в следующей теореме.

<u>ТЕОРЕМА</u>. Любое отношение частичного порядка – линеаризуемо.

На рис. 2.6, *а* приведен пример отношения порядка, а на рис. 2.6, *б* – одна из допустимых его линеаризаций в виде диаграммы Хассе.





а) частичный порядок R_1 б) линеаризация R_2

Рис. 2.6. Отношение частичного порядка (а) и его линеаризация (б)

Необходимо отметить, что задача поиска линеаризации заданного отношения частичного порядка не имеет, как правило, однозначного решения.

• Задачи и упражнения

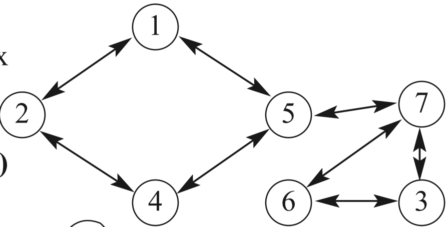
① Доказать, что $(A \cup B) \times C = (A \times C) \cup (B \times C)$.

② Определить, сколько различных бинарных отношений $R (R \subseteq A \times A)$ и различных бинарных отношений Q $(Q \subseteq A \times B)$ можно построить, если $|A| = n, |B| = m$.

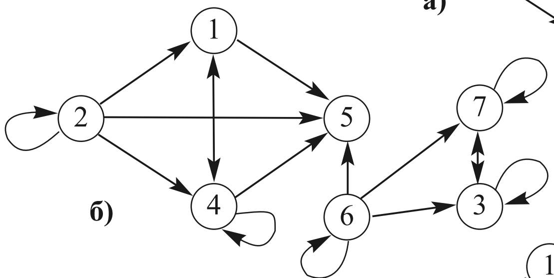
③ Определить, сколько различных одноместных функций $F: A \to B$ можно построить, если $|A| = n, |B| = m$.

④ Определить, сколько различных одноместных взаимно обратных функций $F: A \to B$ можно построить, если $|A| = n, |B| = n$.

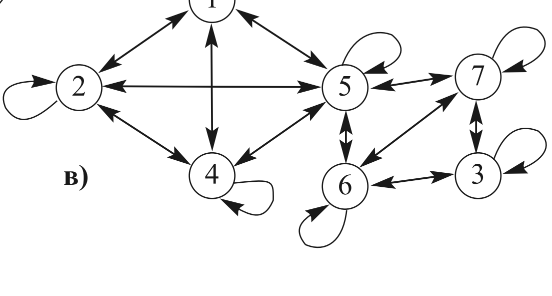
⑤ Определить свойства следующих отношений, заданных в графическом виде:



а)



б)



в)

⑥ Для заданных в предыдущей задаче отношений построить результат выполнения следующих операций:

- R_1^{-1};
- $R_1 \cap R_3$;
- $\overline{R}_3$;
- $R_2 \cap \overline{R}_3$.

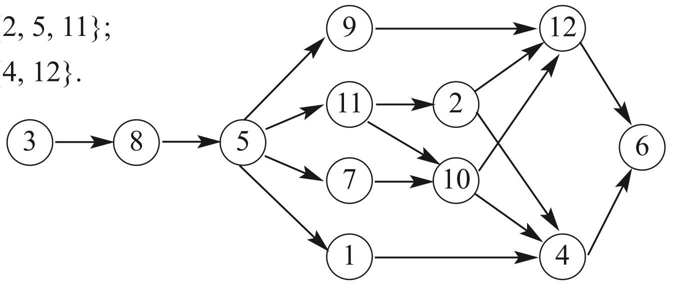
⑦ На множестве $M=\{x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6,\ x_7,\ x_8\}$ задать произвольное бинарное отношение, которое обладает следующим набором свойств:

a) рефлексивность, антисимметричность и транзитивность;
b) рефлексивность, не симметричность и транзитивность;
c) симметричность и антисимметричность;
d) транзитивность и интранзитивность;
e) не рефлексивность, не симметричность и транзитивность.

Для задания отношений использовать графический способ, матричный и функциональный способы.

⑧ Для множества M={1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11} с заданным на нем посредством диаграммы Хассе отношением порядка определить экстремальные характеристики (максимальные и минимальные элементы, мажоронты и миноранты, наибольшие и наименьшие элементы, супремум и инфинум) подмножества M', если:

- $M_1'=\{1, 2, 9\}$;
- $M_2'=\{2, 5, 11\}$;
- $M_3'=\{4, 12\}$.



3. МАТЕМАТИЧЕСКАЯ ЛОГИКА. ЛОГИКА ВЫСКАЗЫВАНИЙ

3.1. Логическое высказывание и его свойства

Основным математическим объектом, который исследуется в логике высказываний, является «*логическое высказывание*». Это понятие относится к фундаментальным понятиям в математике. И, как следствие этого, подобно множествам, оно не имеет конструктивного определения. В качестве интуитивного определения можно привести следующее.

> ***Логическое высказывание*** – это повествовательное предложение, относительно которого со всей очевидностью можно утверждать, что оно либо *истинно*, либо *ложно*.

Из этого определения следует:

- логическое высказывание – это некоторое утверждение, выраженное на естественном (русском, английском или другом) языке;
- логическое высказывание не может быть вопросительным или восклицательным предложением;
- логическое высказывание имеет некоторый содержательный смысл (семантику), посредством которого может быть установлена мера его истинности (истина/ложь);
- логическое высказывание, являясь, по сути, объектом, предполагает наличие субъекта, который способен оценить или задать его истинность.

Примеры:

– *Москва – столица Российской Федерации*. Это высказывание *истинно*.

– *Нью-Йорк – столица США*. Это высказывание *ложно*.

– *25>34*. Это высказывание *ложно*.

– *Сегодня ясная погода*. Это выражение является логическим высказыванием, но мера его истинности может быть определена *субъектом*, который в соответствии с реальной погодной обстановкой объявит его истинным либо ложным.

– *Город стоит на берегу реки*. Это выражение не является логическим высказыванием: невозможно определить его истинность, так как нет указания, о каком городе и о какой реке идет речь.

– *X>24*. Выражение не является логическим высказыванием. Для него нельзя установить значение истинности.

– *Амадей Моцарт был отравлен его другом Антонио Сальери*. Это выражение не является логическим высказыванием, так как историкам и исследователям (субъектам) не удалось установить меру его истинности.

Приведенные примеры демонстрируют определенные сложности, которые возникают при исследовании логических высказываний. Первая из них связана с тем, что естественный язык обладает высокой выразительной силой и принципиально не может быть сведен к простым логическим формализмам и схемам. Другая сложность связана с отсутствием конструктивного определения логического высказывания. Следствием этого являются логические парадоксы, которые были исследованы Расселом.

Вот один из них. «*Я – лжец*», – сказал лжец (лжец – это человек, который никогда не говорит правду). Какова мера истинности этого высказывания? Если предположить, что значение выражения – истина, тогда оказывается, что лжец сказал правду. Но в таком случае он не лжец. А если он не лжец, тогда он солгал. И, следовательно, высказывание – ложно. Но если высказывание ложно, тогда говорящий не является лжецом. А если говорящий – не лжец, тогда он говорит неправду. Круг замкнулся. Одно и то же высказывание является одновременно ложным и истинным, что недопустимо в логике высказываний.

В логике высказываний отвлекаются от *содержания*, *семантики* логических высказываний, сосредоточивая основное внимание на мере их истинности: *истина* и *ложь*. В математической литературе, в языках программирования для обозначения меры истинности часто используются следующие синонимы:

ИСТИНА – True, T, И, Да, 1;

ЛОЖЬ – False, F, Л, Нет, 0.

Далее по тексту для краткости вместо термина «логическое высказывание» будет использоваться просто «высказывание», а истинность высказывания будет обозначаться как 1 (истина) либо 0 (ложь).

Высказывания бывают простыми и сложными. Простые высказывания нельзя разделить на части без потери свойства «быть высказыванием» для каждой из частей. Сложные высказывания образуются из простых посредством специальных логических связок (операций). К таким логическим связкам относятся *логическое сложение*, *логическое умножение*, *отрицание*, *логическое следование*, *эквивалентность* и *сложение Жегалкина*. Эти операции широко используются в математических, юридических, инструктивных и других текстах, в которых требуется определенным образом обеспечить соотношение истинности тех или иных фактов.

Ниже рассмотрены эти операции и проведено вычисление истинности сложных высказываний, а также дана содержательная интерпретация логических операций в естественном языке.

Пусть a и b – некоторые простые высказывания.

Операция логического сложения – «$\vee$» (синонимы – *дизъюнкция*, OR, +, ИЛИ, MAX).

Сложное высказывание «$a \vee b$» интерпретируется в естественном языке обычно как:

– a или b;
– a либо b, либо оба.

Вычисление истинности этого высказывания в зависимости от истинности образующих его высказываний задается таблицей (таблицей истинности).

a	b	$a \vee b$
0	0	0
0	1	1
1	0	1
1	1	1

Как видно из таблицы, сложное высказывание «a или b» ложно тогда, когда оба образующие его высказывания ложны. Во всех остальных случаях оно истинно.

Операция логического умножения – «$\wedge$» (синонимы – *конъюнкция*, AND, *, &, И, MIN).

Сложное высказывание «$a \wedge b$» интерпретируется в естественном языке обычно как:

– a и b;
– известно, что:
a;
b.

Вычисление истинности этого высказывания в зависимости от истинности образующих его высказываний задается таблицей.

a	b	$a \wedge b$
0	0	0
0	1	0
1	0	0
1	1	1

Как видно из таблицы, сложное высказывание «a и b» истинно тогда, когда оба образующие его высказывания истинны. Во всех остальных случаях оно ложно.

Операция отрицания – «$\overline{\ \ }$» (синонимы – *инверсия*, NO, NOT, НЕ, ¬).

Высказывание «$\overline{a}$» интерпретируется в естественном языке обычно как:

– не a;
– неверно, что a.

Вычисление истинности этого высказывания в зависимости от истинности высказывания a задается таблицей.

a	$\overline{a}$
0	1
1	0

Как видно из таблицы, высказывание «не a» изменяет значение истинности высказывания a на противоположное.

Операция логического следования – «$\rightarrow$» (синонимы – *импликация*, $\leq$).

Сложное высказывание «$a \rightarrow b$» интерпретируется в естественном языке обычно как:

– если a, то b;
– b, если a;
– из a следует b;
– b следует из a;
– a – достаточное условие для b;
– a, если только b;
– b – необходимое условие для a;
– a. Вывод: b;
– a имплицирует b;
– a – причина, b – следствие и т.п.

a	b	$a \rightarrow b$
0	0	1
0	1	1
1	0	0
1	1	1

Вычисление истинности этого высказывания в зависимости от истинности образующих его высказываний задается таблицей.

Как видно из таблицы, сложное высказывание «если a, то b» ложно тогда, когда из истины следует ложь. Во всех остальных случаях оно истинно.

<u>Замечание</u>. В математике очень часто в теоремах используется оборот «если…, то…». Доказательство таких теорем требует установление факта, что из истины следует истина: $1 \rightarrow 1 = 1$. Если этот факт установлен (теорема доказана), то возможность ее применения к решению конкретной задачи требует лишь определения истинности левой части импликации.

Эквивалентность – «$\leftrightarrow$» (синонимы – *эквиваленция*, $=$, $\approx$, $\equiv$).

Сложное высказывание «$a \leftrightarrow b$» интерпретируется в естественном языке обычно как:

– *a* эквивалентно *b*;
– *a,* если и только если *b*;
– *a,* тогда и только тогда, когда *b*;
– *a* – необходимое и достаточное условие для *b*.

Вычисление истинности этого высказывания в зависимости от истинности образующих его высказываний задается таблицей.

a	b	$a \leftrightarrow b$
0	0	1
0	1	0
1	0	0
1	1	1

Как видно из таблицы, сложное высказывание «*a* эквивалентно *b*» истинно тогда и только тогда, когда совпадает истинность образующих его высказываний *a* и *b*. Во всех остальных случаях оно ложно.

Сложение Жегалкина – «$\oplus$» (синонимы – *сложение по модулю* 2, XOR, ЛИБО, $\neq$).

Сложное высказывание «$a \oplus b$» интерпретируется в естественном языке обычно как:

– *a* либо *b*;
– *a* или *b*, но не оба.

a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Вычисление истинности этого высказывания в зависимости от истинности образующих его высказываний задается таблицей.

Как видно из таблицы, сложное высказывание «*a* либо *b*» истинно тогда и только тогда, когда не совпадают истинности образующих его высказываний *a* и *b*. Во всех остальных случаях оно ложно.

В математической логике сложные логические выражения записываются в виде формул: каждому простому высказыванию ставится в соответствие значение его истинности, логические связки заменяются соответствующими символами логических операций, а скобки поддерживают необходимые структуры. Для введенных логических операций определено отношение старшинства (приоритет). Старшей, естественно, является операция, стоящая в скобках. При этом операция отрицания, по приоритету, приравнивается к скобкам. Затем, в порядке уменьшения приоритета, идет операция логического умножения, далее – операция логического сложения. Для операций логического следования, эквивалентности и сложения Жегалкина установлен самый низкий приоритет, а старшинство между ними регулируется в общем случае скобками. Так, выражение $0\to1\to0$ является некорректным. В зависимости от того, в каком порядке будут выполняться операции, получаются разные результаты: $(0\to1)\to0=1\to0=0$ или $0\to(1\to0)= 0\to0=1$.

Задачу, в результате решения которой некоторому логическому суждению на естественном языке ставится в соответствие логическая формула, называют *формализацией*. Хотя в общем случае не существует универсальной процедуры формализации логических высказываний, можно рекомендовать следующую схему для ее решения:

1. Выделить простые высказывания и заменить их некоторыми символами.
2. Выделить логические связки.
3. Построить логическую структуру выражения в виде дерева. Каждой внутренней вершине дерева ставится в соответствие логическая связка (операция). Каждой висячей вершине ставится в соответствие символ простого высказывания (рис. 3.1).
4. Получить формулу посредством обхода дерева от нижнего левого узла к корню дерева с учетом старшинства логических операций.
5. Подставить вместо символов, обозначающих простые высказывания, значения их истинности.

В качестве внутренних вершин при построении логической структуры используются фрагменты дерева, приведенные на рис. 3.1.

Каждая из этих вершин, за исключением отрицания, имеет два входа (для отрицания – один) и один выход. Значение выхода узла определяется значениями его входов и его типом (логической связкой). Вход каждого узла может быть соединен не более чем с одним

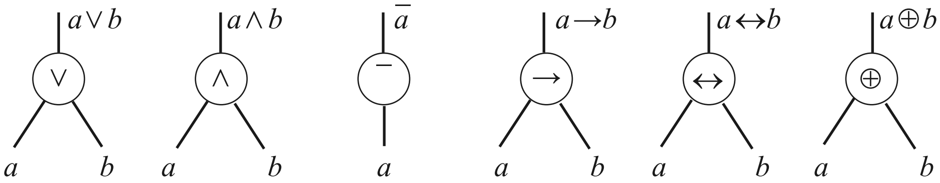


Рис. 3.1. Типы внутренних вершин логических структур

выходом другого, что позволяет строить древовидные структуры сложных логических высказываний. Корню дерева соответствует искомая формула, а его висячим (свободным) входам соответствуют простые высказывания.

На рис. 3.2 приведен пример логической структуры, которой соответствует следующее сложное высказывание: «Неверно, что если не *a*, то *b* и *c*». Если при этом *a* – истинно, а *b* и *c* – ложны, это высказывание $\overline{\overline{a}\to b\wedge c}=\overline{\overline{1}\to 0\wedge 0}=\overline{0\to 0}=\overline{1}=0$ является ложным.

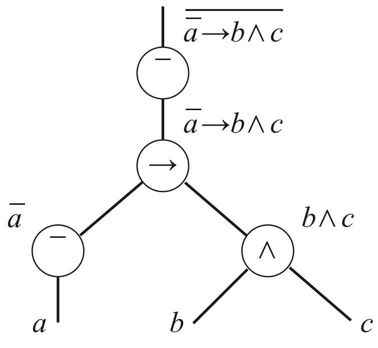


Рис. 3.2. Пример логической структуры

<u>Примеры сложных логических высказываний</u> (суждений):

– Целое число делится на 6 тогда и только тогда, когда оно делится на 2 и делится на 3.

– Необходимым условием деления целого числа на 6 является его делимость на 2.

– Олимпиаду по программированию выиграл студент Иванов либо студент Петров.

– Неверно, что если погода пасмурная, то идет дождь и дует ветер.

– Если уменьшить издержки производства или повысить производительность труда, то повысится рентабельность.

Пусть для приведенных в примере сложных логических высказываний введены следующие обозначения простых высказываний и определены значения их истинности (табл. 3.1).

Таблица 3.1

Примеры простых высказываний

Символ	Простое высказывание	Мера истинности простого высказывания
a	целое число делится на 2	1
в	целое число делится на 3	0
c	целое число делится на 6	1
p	олимпиаду по программированию выиграл Петров	0
i	олимпиаду по программированию выиграл Иванов	1
r	ясная погода	1
t	ветреная погода	0
s	дождливая погода	1

Тогда следующие формулы являются символической записью (формализацией) первых четырех сложных высказываний. Рядом показано значение истинности.

$c\leftrightarrow a\wedge b$, $\quad 1\leftrightarrow 1\wedge 0=0$;

$c\to a$, $\quad 1\to 1=1$;

$p\oplus i$, $\quad 0\oplus 1=1$;

$\overline{\overline{r}\to s\wedge t}$, $\quad \overline{\overline{1}\to 1\wedge 0}=0$.

<u>Замечание</u>. Логические формулы, получаемые в результате формализации суждений, в общем случае логическими высказываниями

не являются, так как значение истинности образующих формулу простых высказываний не определено. Такие логические формулы часто называют *высказывательными функциями*. При этом высказывательная функция превращается в логическое высказывание, если в нее подставить значения истинности образующих ее простых высказываний. Такую подстановку называют *интерпретацией*.

Введенные логические операции позволяют решать две основные задачи логики высказываний:

1) проводить формализацию логических высказываний (суждений);

2) вычислять истинность сложных высказываний при заданных значениях истинности образующих их простых высказываний.

Вычисление истинности (интерпретация) сложных высказываний проводится двумя способами:

- подстановкой конкретных значений высказываний непосредственно в формулу;
- с помощью таблиц истинности (ТИ).

Таблицу истинности применяют тогда, когда есть необходимость вычислить истинность высказывательной функции при всех возможных значениях истинности образующих ее простых высказываний (провести интерпретацию).

Столбцы ТИ (рис. 3.3) разделяются на две части: аргументную и операционную. Каждому столбцу в аргументной части ставится в соответствие простое высказывание (a, b, c). Каждому столбцу в операционной части ставится в соответствие логическая операция высказывательной функции в порядке ее вычисления ($ОП_1$, $ОП_2$, …, $ОП_k$).

Каждой строке ТИ в аргументной части ставится в соответствие распределение истинностей исходных простых высказываний. В этой же строке ТИ в операционной части приводятся результаты вычисления соответствующих операций.

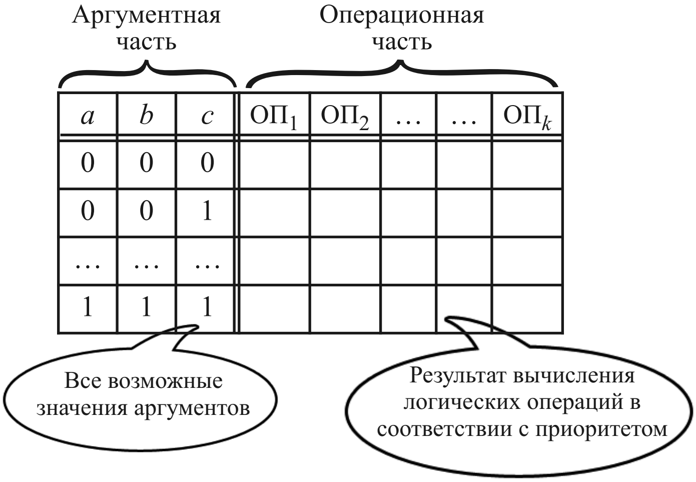


Рис. 3.3. Таблица истинности

Ниже, в качестве примера приведено решение следующей задачи. При исследовании причины сбоев в работе системы управления удаленным объектом эксперт пришел к выводу, что их причиной могут быть сбои в работе датчиков x_1, x_2 и x_3. В качестве заключения он сформулировал следующие утверждения:

- если сбоит датчик x_1, тогда датчик x_2 работает без сбоев;
- из двух датчиков x_2 и x_3 только один работает правильно;
- если датчик x_3 работает без сбоев, тогда сбоит датчик x_1;
- достаточным условием для правильной работы датчика x_3 является работа без сбоев датчика x_1 и датчика x_2.

Требуется на основании заключения эксперта определить (вычислить), какие датчики работают без сбоя, а какие – с нарушениями.

Решение этой задачи предполагает:

1) проведение формализации заключения эксперта (получение высказывательной функции);

2) проведение интерпретации высказывательной функции с целью определения области ее истинности.

Формализация:

пусть a_1 есть простое логическое высказывание «датчик x_1 работает без сбоев»;

пусть a_2 есть простое логическое высказывание «датчик x_2 работает без сбоев»;

пусть a_3 есть простое логическое высказывание «датчик x_3 работает без сбоев».

Отрицание высказывания a_1 будет означать, что датчик x_1 работает со сбоем. Подобное рассуждение относится и к a_2, и к a_3.

Первое высказывание эксперта формализуется как $(\overline{a}_1 \rightarrow a_2)$.

Второе высказывание эксперта формализуется как $(a_2 \oplus a_3)$.

Третье высказывание эксперта формализуется как $(a_3 \rightarrow \overline{a}_1)$.

Четвертое высказывание эксперта формализуется как $(a_1 \wedge a_2 \rightarrow a_3)$.

С учетом того, что мнение эксперта – это истина, все его суждения истинны, и, следовательно, они должны быть связаны операцией логического умножения: $(\overline{a}_1 \rightarrow a_2) \wedge (a_2 \oplus a_3) \wedge (a_3 \rightarrow \overline{a}_1) \wedge (a_1 \wedge a_2 \rightarrow a_3)$. Это и есть искомая высказывательная функция.

Таблица 3.2

Таблица истинности

a_1	a_2	a_3	① $\overline{a}_1 \rightarrow a_2$	② $a_2 \oplus a_3$	③ $a_3 \rightarrow \overline{a}_1$	④ $a_1 \wedge a_2 \rightarrow a_3$	①∧②∧③∧④
0	0	0	0	0	1	1	0
0	0	1	0	1	1	1	0
0	1	0	1	1	1	1	1
0	1	1	1	0	1	1	0
1	0	0	1	0	1	1	0
1	0	1	1	1	0	1	0
1	1	0	1	1	1	0	0
1	1	1	1	0	0	1	0

По условию задачи требуется определить, при каких значениях высказываний a_1, a_2 и a_3 обеспечивается истинность высказывательной функции.

Из таблицы истинности (табл. 3.2) следует, что мнение эксперта истинно только тогда, когда высказывания a_1 и a_3 ложны, а a_2 – истинно. В соответствии с условиями задачи это означает, что датчики x_1 и x_3 неисправны, работают со сбоями, а датчик x_2 – исправен.

3.2. Логические функции и алгебра логики

Пусть $x_1, x_2, \ldots, x_n$ – множество логических переменных, каждая из которых может принимать значения из множества $\{0, 1\}$. По определению, логическая функция от n-переменных

$f(x_1, x_2, \ldots, x_n)$: $\{0,1\}^n \to \{0,1\}$.

Пусть $\sigma_1, \sigma_2, \ldots, \sigma_n$ – фиксированный набор значений переменных $x_1, x_2, \ldots, x_{\mathrm{n}}$ логической функции $f(x_1, x_2, \ldots, x_n)$. Тогда $f(\sigma_1, \sigma_2, \ldots, \sigma_n)$ есть значение этой функции при заданном наборе $\sigma_1, \sigma_2, \ldots, \sigma_n$ значений переменных $x_1, x_2, \ldots, x_n$ соответственно. Далее в этом разделе для краткости вместо понятия «*логическая функция*» будет использоваться понятие «*функция*».

Для задания логической функции используется табличный способ или аналитический, с помощью логической формулы. При табличном способе аргументная часть таблицы задает все возможные значения переменных $x_1, x_2, \ldots, x_n$, а правый столбец определяет соответствующее значение функции.

Аргументная часть				Значения функции
x_1	x_2	…	x_n	$f(x_1, x_2, \ldots, x_n)$
0	0	…	0	$f(0, 0, \ldots, 0)$
0	0	…	1	$f(0, 0, \ldots, 1)$
…	…	…	…	…
1	1	…	1	$f(1, 1, \ldots, 1)$

Пусть задана функция $f(x_1, x_2, \dots, x_n)$, и x_i – ее некоторая переменная.

Функция $f(x_1, x_2, \dots, x_n)$ *существенно зависит* от переменной x_i, $1<i<n$, если существует набор $\sigma_1, \sigma_2, \dots, \sigma_{i-1}, \sigma_{i+1}, \dots, \sigma_n$ значений переменных $x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_n$, при котором $f(\sigma_1, \sigma_2, \dots, \sigma_{i-1}, 0, \sigma_{i+1}, \sigma_{i+2}, \dots, \sigma_n) \neq f(\sigma_1, \sigma_2, \dots, \sigma_{i-1}, 1, \sigma_{i+1}, \sigma_{i+2}, \dots, \sigma_n)$. В противном случае переменная x_i считается несущественной, фиктивной переменной, и функция $f(x_1, x_2, \dots, x_n)$ не зависит от нее.

Введение в рассмотрение фиктивных переменных позволяет трактовать любую функцию как функцию некоторой фиксированной размерности n, не ниже требуемой. В таблице приведены примеры двух функций $f(x_1, x_2, x_3)$ и $g(x_1, x_2, x_3)$, первая из которых существенно зависит от всех переменных, а вторая не зависит от переменной x_2.

x_1	x_2	x_3	$f(x_1, x_2, x_3)$	$g(x_1, x_2, x_3)$
0	0	0	1	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0

Если значения функции $f(x_1, x_2, \dots, x_n)$ определены для всех возможных значений $\sigma_1, \sigma_2, \dots, \sigma_n$ переменных $x_1, x_2, \dots, x_n$, функция $f(x_1, x_2, \dots, x_n)$ является полностью определенной. Если это не так, функция считается частично определенной. В качестве примера рассмотрим модель системы управления обогревателями (рис. 3.4).

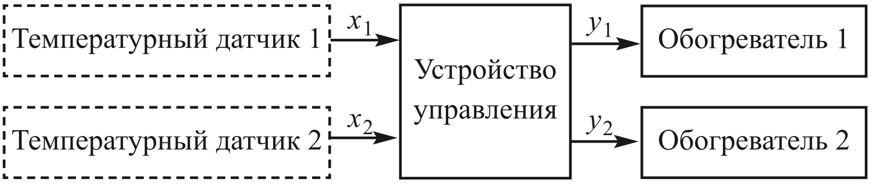


Рис. 3.4. Модель устройства управления обогревателями

Температурные датчики 1 и 2 контролируют температуру, при этом:

$$x_1 = \begin{cases} 1, \text{ если } t° < -10° \text{ C;} \\ 0 \text{ в противном случае;} \end{cases}$$

$$x_2 = \begin{cases} 1, \text{ если } t° < -20° \text{ C;} \\ 0 \text{ в противном случае.} \end{cases}$$

Включение и выключение обогревателей осуществляется управляющими логическими сигналами y_1 и y_2:

$$y_1 = \begin{cases} 1, \text{ включен обогреватель 1;} \\ 0, \text{ обогреватель 1 выключен;} \end{cases}$$

$$y_2 = \begin{cases} 1, \text{ включен обогреватель 2;} \\ 0, \text{ обогреватель 2 выключен.} \end{cases}$$

Алгоритм управления обогревателями состоит в следующем:
если $-10° \leq t°$, обогреватели должны быть отключены;
если $-20° \leq t° < -10°$, должен быть включен первый обогреватель;
если $t° < -20°$, должны быть включены оба обогревателя.

Ниже приведены логические функции $y_1(x_1, x_2)$ и $y_2(x_1, x_2)$ управления включением и выключением обогревателями в зависимости от значений переменных x_1, x_2.

x_1	x_2	$y_1(x_1, x_2)$	$y_2(x_1, x_2)$
0	0	0	0
0	1	–	–
1	0	1	0
1	1	1	1

Как видно из таблицы, значение управляющих функций $y_1(x_1, x_2)$ и $y_2(x_1, x_2)$ не определено, когда

x_1=1 и x_2=0. Эти значения физически не реализуемы, так как температура не может быть одновременно больше –10° и меньше –20°. Функции $y_1(x_1, x_2)$ и $y_2(x_1, x_2)$ – частично определенные.

Пусть $\Im(x_1, x_2, \ldots, x_n)$ – множество всех логических функций от n переменных.

Совокупность

$\langle \Im(x_1, x_2, \ldots, x_n), \vee, \wedge, {}^{-} \rangle$

называют n-основной *алгеброй логики*, если для произвольных функций φ, ψ и λ из $\Im(x_1, x_2, \ldots, x_n)$ справедливы следующие свойства:

– свойство *идемпотентности*:

$\varphi \vee \varphi = \varphi$, $\quad \varphi \wedge \varphi = \varphi$;

– свойство *коммутативности*:

$\varphi \vee \psi = \psi \vee \varphi$, $\quad \varphi \wedge \psi = \psi \wedge \varphi$;

– свойство *ассоциативности*:

$\varphi \vee \psi \vee \lambda = (\varphi \vee \psi) \vee \lambda$,

$\varphi \wedge \psi \wedge \lambda = (\varphi \wedge \psi) \wedge \lambda$;

– свойство *дистрибутивности*:

$\varphi \wedge (\psi \vee \lambda) = \varphi \wedge \psi \vee \varphi \wedge \lambda$,

$\varphi \vee \psi \wedge \lambda = (\varphi \vee \psi) \wedge (\varphi \vee \lambda)$;

– свойство *де-Моргана*:

$\overline{\varphi \vee \psi} = \overline{\varphi} \wedge \overline{\psi}$,

$\overline{\varphi \wedge \psi} = \overline{\varphi} \vee \overline{\psi}$;

– свойство *двойного отрицания*: $\overline{\overline{\varphi}} = \varphi$;

– свойства 0 и 1:

$\varphi \wedge 1 = \varphi$, $\quad \varphi \vee 1 = 1$,

$\varphi \wedge 0 = 0$, $\quad \varphi \vee 0 = \varphi$,

$\varphi \wedge \overline{\varphi} = 0$, $\quad \varphi \vee \overline{\varphi} = 1$.

К дополнительным свойствам алгебры логики относятся свойство поглощения и свойство склеивания, которые могут быть получены из предыдущих свойств.

– *свойство поглощения;*

$\varphi \vee \varphi \wedge \psi = \varphi$;

$\varphi \wedge (\varphi \vee \psi) = \varphi$;

– *свойство склеивания;*

$\varphi \wedge \psi \vee \varphi \wedge \overline{\psi} = \varphi$;

$(\varphi \vee \psi) \wedge (\varphi \vee \overline{\psi}) = \varphi$.

Введенная алгебра и свойства операций позволяют решать задачи эквивалентного преобразования логических формул.

Примеры эквивалентных преобразований:

1) $x \vee \overline{x} \wedge y = (x \vee \overline{x}) \wedge (x \vee y) = 1 \wedge (x \vee y) = x \vee y$.

2) $x \wedge (x \vee y) = (x \vee 0) \wedge (x \vee y) = x \vee 0 \wedge y = x \vee 0 = x$ (свойство поглощения).

3) Доказать, что $x \wedge y \vee \overline{x} \wedge \overline{z} \vee y \wedge \overline{z} = x \wedge y \vee \overline{x} \wedge \overline{z}$.

Выполнив преобразование элемента левой части

$y \wedge \overline{z} = 1 \wedge y \wedge \overline{z} = (x \vee \overline{x}) \wedge y \wedge \overline{z} = x \wedge y \wedge \overline{z} \vee \overline{x} \wedge y \wedge \overline{z}$,

подставим этот результат в исходное выражение. Применяя свойства коммутативности и ассоциативности, а затем и свойство поглощения, приводим выражение к требуемому виду:

$x \wedge y \vee \overline{x} \wedge \overline{z} \vee \overline{x} \wedge y \wedge \overline{z} \vee \overline{x} \wedge y \wedge \overline{z} =$

$= (x \wedge y \vee x \wedge y \wedge \overline{z}) \vee (\overline{x} \wedge \overline{z} \vee \overline{x} \wedge \overline{z} \wedge y) = x \wedge y \vee \overline{x} \wedge \overline{z}$,

что и требовалось доказать.

В связи с тем, что логическая функция может быть задана таблично либо аналитически, возможны две задачи.

Задача 1. Задана логическая функция в виде формулы алгебры логики. Требуется построить табличное представление этой функции.

Задача 2. Задана логическая функция в табличном виде. Требуется найти аналитическое (формульное) представление этой функции.

Решение первой задачи сводится к построению таблицы истинности исходной функции, в которую из операционной части выносится только заключительный столбец.

Пример решения первой задачи. Требуется построить табличное представление функции $f(x_1, x_2, x_3)=\overline{x}_1\wedge(\overline{x}_2\vee\overline{x}_3)\vee x_2\wedge x_3$.

Таблица 3.3

Табличное представление функции $f(x_1, x_2, x_3)$

x_1	x_2	x_3	$\overline{x}_1\wedge(\overline{x}_2\vee\overline{x}_3)$	$x_2\wedge x_3$	$f(x_1, x_2, x_3)$
0	0	0	1	0	1
0	0	1	1	0	1
0	1	0	1	0	1
0	1	1	0	1	1
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	1	1

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Решение. В табл. 3.3 приведены таблицы истинности для заданной функции и искомое табличное представление заданной функции.

Вторая задача не имеет однозначного решения. Действительно, при аналитическом представлении одна и та же функция может представляться различными формулами, которые эквивалентны между собой. В частности, рассмотренную выше в примере функцию $f(x_1, x_2, x_3)$ можно представить как $\overline{x}_1\vee x_2\wedge x_3$. Установление этого факта предоставляется читателю в качестве упражнения.

3.3. Нормальные формы

Пусть $x_1, x_2, \ldots, x_n$ – множество логических переменных.

По определению, ***первичный терм*** переменной x_i есть

$$x_i^{\sigma_i}=\begin{cases} x_i, \text{ если } \sigma_i=1; \\ \overline{x}_i, \text{ если } \sigma_i=0. \end{cases}$$

Первичный терм либо конъюнкция некоторого числа первичных термов различных переменных называется ***импликантой.***

Импликанта, в образовании которой участвуют первичные термы всех n переменных, называется ***конституентой***.

Для логических функций справедлива теорема о разложении функции по переменным.

ТЕОРЕМА *о разложении*. Любая отличная от тождественного нуля логическая функция $f(x_1, x_2, \ldots, x_n)$ от n ($n\geq 1$) переменных может быть представлена в виде разложения:

$$f(x_1, x_2, \ldots, x_k, \ldots, x_n)= \bigvee_{\sigma_1,\sigma_2,\ldots,\sigma_k} x_1^{\sigma_1}\wedge x_2^{\sigma_2}\wedge\ldots\wedge x_k^{\sigma_k}\wedge f(\sigma_1, \sigma_2, \ldots, \sigma_k, x_{k+1}, \ldots, x_n).$$

Функции $f(\sigma_1, \sigma_2, \ldots, \sigma_k, x_{k+1}, \ldots, x_n)$ называют *остаточными функциями*.

Следствие 1. Разложение отличной от тождественного нуля логической функции по i-й переменной равно:

$$f(x_1, x_2, \ldots, x_i, \ldots, x_n)=\overline{x}_i\wedge f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n)\vee x_i\wedge \wedge f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n).$$

Следствие 2. Предельное разложение логической функции по n переменным есть дизъюнкция конституент:

$$f(x_1, x_2, \ldots, x_k, \ldots, x_n)= \bigvee_{(\sigma_1,\sigma_2,\ldots,\sigma_n):f(\sigma_1,\sigma_2,\ldots,\sigma_n)=1} x_1^{\sigma_1}\wedge x_2^{\sigma_2}\wedge\ldots\wedge x_n^{\sigma_n}.$$

Разложение функции по всем переменным называют *совершенной дизъюнктивной нормальной формой*, или кратко – СовДНФ. Термин «форма» здесь следует понимать как определенный тип формульного (аналитического) представления функции. Термин «нормальная» означает, что в представлении функции отсутствуют скобки, а отрицания используются только в первичных термах. Термин «дизъюнктивная» определяется тем, что форма имеет регулярное представление в виде дизъюнкции конституент, для которых функция $f(x_1, x_2, \ldots, x_n)$ принимает значение 1. Термин «совершенная» означает, что эта форма для функции является единственной с точностью до перестановки конституент.

Основное свойство конституенты заключается в том, что она, как логическая функция, принимает значение 1 только на одном единственном наборе значений переменных, который определяется степенями первичных термов. На всех остальных наборах она принимает значение 0. Конституента $x_1^0 \wedge x_2^1 \wedge x_3^0 = \overline{x}_1 \wedge x_2 \wedge \overline{x}_3$ принимает значение 1 только тогда, когда $x_1=0$, $x_2=1$, $x_3=0$, так как $\overline{0} \wedge 1 \wedge \overline{0} = 1$. Подстановка любых других значений переменных в эту конституенту приводит к 0. Именно это определяет тот факт, что в СовДНФ включены только те конституенты, на двоичных наборах значений переменных которых функция принимает значение 1. Для функции трех переменных, заданной таблично (табл. 3.4), соответствующая ей СовДНФ есть

$$f(x_1, x_2, x_3)_{\text{СовДНФ}}=$$
$$=\overline{x}_1 \wedge \overline{x}_2 \wedge \overline{x}_3 \vee \overline{x}_1 \wedge \overline{x}_2 \wedge x_3 \vee \overline{x}_1 \wedge x_2 \wedge \overline{x}_3 \vee \overline{x}_1 \wedge x_2 \wedge x_3 \vee x_1 \wedge x_2 \wedge x_3.$$

Таблица 3.4

Представление функции $f(x_1, x_2, x_3)$

	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
✓	0	0	0	1
✓	0	0	1	1
✓	0	1	0	1
✓	0	1	1	1
	1	0	0	0
	1	0	1	0
	1	1	0	0
✓	1	1	1	1

Слева от таблицы отмечены строки, которым соответствуют конституенты, вошедшие в СовДНФ (для которых $f(x_1, x_2, x_3)=1$).

Лемма 1. Дизъюнкция всех конституент тождественно равна 1.

Доказательство. Из свойств алгебры логики справедливы тождества:

$$\&\begin{cases} x_1^0 \vee x_1^1 = 1; \\ \dots \\ x_i^0 \vee x_i^1 = 1; \\ \dots \\ x_n^0 \vee x_n^1 = 1. \end{cases} \qquad (1^*)$$

Конъюнкция правых и левых частей равенств (1*) приводит к выражению $(x_1^0 \vee x_1^1) \wedge \ldots \wedge (x_i^0 \vee x_i^1) \wedge \ldots \wedge (x_n^0 \vee x_n^1) = 1$. После раскрытия скобок по свойству дистрибутивности получается дизъюнкция всех конституент:

$$(x_1^0 \vee x_1^1) \wedge \ldots \wedge (x_i^0 \vee x_i^1) \wedge \ldots \wedge (x_n^0 \vee x_n^1) =$$
$$= C_1 \vee C_2 \vee \ldots \vee C_j \vee \ldots \vee C_m = 1, \qquad (2^*)$$

где $C_j = x_1^{\sigma_{1j}} \wedge x_2^{\sigma_{2j}} \wedge \ldots \wedge x_n^{\sigma_{nj}}$ является j-й конституентой $(1 \leq j \leq m)$.

<u>Лемма 2</u>. Любой первичный терм равен дизъюнкции конституент, содержащих этот терм.

<u>Доказательство</u>. В предыдущей лемме показано, что дизъюнкция всех конституент равна 1. Конъюнкция правой и левой частей выражения (2*) с первичным термом $x_i^{\sigma_i}$ приводит к

$$C_1 \wedge x_i^{\sigma_i} \vee C_2 \wedge x_i^{\sigma_i} \vee \ldots \vee C_j \wedge x_i^{\sigma_i} \vee \ldots \vee C_m \wedge x_i^{\sigma_i} = x_i^{\sigma_i}. \qquad (3^*)$$

Анализ конъюнкции $C_j \wedge x_i^{\sigma_i}$ $(1 \leq j \leq m)$ показывает, что она равна либо самой конституенте C_j, либо 0:

$$C_j \wedge x_i^{\sigma_i} = x_1^{\sigma_{1j}} \wedge \ldots \wedge x_i^{\sigma_{ij}} \wedge \ldots \wedge x_n^{\sigma_{nj}} \wedge x_i^{\sigma_i} = \begin{cases} C_j, \text{ если } \sigma_{ij} = \sigma_i; \\ 0 \text{ в противном случае.} \end{cases}$$

А это означает, что в выражении (3*) останутся только те конституенты, которые содержат терм $x_i^{\sigma_i}$.

<u>Лемма 3</u>. Число различных конституент (m) не превышает величины 2^n.

<u>Доказательство</u>. Действительно, мощность множества конституент совпадает с мощностью множества двоичных наборов длины n, число которых равно 2^n.

Возникает вопрос: почему в условии леммы не стоит только равенство?

Это определяется тем, что в общем случае логические переменные могут быть взаимно зависимыми. Вследствие этого истинность одних переменных определяется истинностью других. А это приводит к уменьшению числа конституент по отношению к величине 2^n.

Лемма 4. Число логических функций от n переменных не превышает величины 2^{2^n}.

Доказательство. Действительно, одна функция от другой отличается набором конституент в ее СовДНФ. Так как верхняя оценка для числа конституент равна 2^n и каждая из конституент может входить или не входить в СовДНФ, число различных СовДНФ совпадает с мощностью множества двоичных векторов длины 2^n, а их число 2^{2^n}.

Совершенная дизъюнктивно нормальная форма является не единственной совершенной формой. Существуют еще и другие формы представления функций, в том числе и нормальные.

Рассмотрим так называемую *совершенную конъюнктивную нормальную форму* (СовКНФ) – конъюнкция некоторого числа стандартных элементов, каждый из которых состоит из дизъюнкций различных первичных термов всех n переменных.

СовКНФ $f(x_1, x_2, ..., x_n)$ может быть получена как инверсия СовДНФ функции $\overline{f}(x_1, x_2, ..., x_n)$, принимающая значение 1 на наборах значений переменных, в которых исходная функция принимает значение 0.

Для примера в табл. 3.5 приведена функция $f(x_1, x_2, x_3)$, для которой выше была построена СовДНФ.

В этой таблице отмечены те строки, в которых функция принимает значение 0 (нулевая область). Тогда $\overline{f}(x_1, x_2, x_3)_{\text{СовДНФ}}=$
$= x_1 \wedge \overline{x}_2 \wedge \overline{x}_3 \vee x_1 \wedge \overline{x}_2 \wedge x_3 \vee x_1 \wedge x_2 \wedge \overline{x}_3.$

Таблица 3.5

	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
	0	0	0	1
	0	0	1	1
	0	1	0	1
	0	1	1	1
✓	1	0	0	0
✓	1	0	1	0
✓	1	1	0	0
	1	1	1	1

А СовКНФ получается как инверсия от $\overline{f}(x_1, x_2, x_3)_{\text{СовДНФ}}$:

$$f(x_1, x_2, x_3)_{\text{СовКНФ}}=\overline{x_1\wedge \overline{x}_2\wedge \overline{x}_3\vee x_1\wedge \overline{x}_2\wedge x_3\vee x_1\wedge x_2\wedge \overline{x}_3}=$$

$$=(\overline{x}_1\vee x_2\vee x_3)\wedge(\overline{x}_1\vee x_2\vee \overline{x}_3)\wedge(\overline{x}_1\vee \overline{x}_2\vee x_3).$$

Совершенные нормальные формы не являются единственной формой аналитического представления логических функций. В частности, в приложениях широко используются классы дизъюнктивных и конъюнктивных нормальных форм (ДНФ и КНФ), которые можно получить из совершенных форм посредством применения к ним свойства склеивания.

Примеры КНФ и ДНФ для рассмотренной функции:

$$f(x_1, x_2, x_3)_{\text{ДНФ}}=\overline{x}_1\wedge \overline{x}_2\vee \overline{x}_1\wedge \overline{x}_3\vee x_2\wedge x_3;$$

$$f(x_1, x_2, x_3)_{\text{КНФ}}=(\overline{x}_1\vee x_2)\wedge(\overline{x}_1\vee x_3).$$

3.4. Задача о нахождении покрытия минимальной стоимости

Одна из практических задач, которая может быть решена с использованием математической логики, – задача о нахождении покрытия минимальной стоимости. Эта задача относится к классу комбинаторных задач дискретной математики, в основе решения которых лежит перебор всех возможных решений. Математическая логика позволяет квалифицированно провести такой перебор.

Постановка задачи.

Дано:

1. Прямоугольная двумерная бинарная таблица (матрица) $T=||t_{ij}||$ размерности $m \times n$. Элемент матрицы $t_{ij} \in \{0, 1\}$.
2. Множество столбцов $Y=\{y_1, y_2, \ldots, y_n\}$.
3. Множество строк $Z=\{z_1, z_2, \ldots, z_m\}$.
4. Вектор стоимостей столбцов $\vec{C}=(c_1, c_2, \ldots, c_n)$.

Если $t_{ij}=1$, то считается, что столбец y_j покрывает строку z_i (или, что эквивалентно, строка z_i покрывается столбцом y_j).

Подмножество столбцов $\pi(\pi \subseteq Y)$ является покрытием строк таблицы T, если:

- для каждой строки из Z найдется столбец из π, ее покрывающий;
- любое собственное подмножество множества π приводит к нарушению предыдущего требования.

Стоимость покрытия π – $C(\pi)$ определяется как сумма стоимостей столбцов, входящих в это покрытие:

$$C(\pi)=\sum_{y_i \in \pi} c_i.$$

Определить: для заданной двоичной таблицы T найти все покрытия строк столбцами, которые характеризуются минимальной стоимостью.

Существуют различные модификации этой задачи. В частности, иногда требуется найти не все, а хотя бы одно покрытие минимальной стоимости. В некоторых случаях, когда стоимости всех столбцов совпадают, поиск покрытий минимальной стоимости сводится к поиску так называемых кратчайших покрытий, т.е. покрытий, содержащих минимальное число столбцов.

Решение задачи. Как отмечалось выше, задача о нахождении покрытия минимальной стоимости относится к классу комбинаторных задач. Ее решение основано на переборе всех возможных решений и сводится к следующей последовательности действий:

1) найти все покрытия двоичной таблицы;
2) вычислить стоимость каждого покрытия;
3) выбрать то или те покрытия, которые имеют минимальную стоимость.

Первый этап в этой схеме наиболее трудоемкий. Для нахождения покрытий двоичной таблицы можно использовать последовательность действий:

- построить логическое условие покрытия всех строк таблицы;
- привести построенное условие к минимальной ДНФ. Каждой импликанте в этой ДНФ будет соответствовать одно покрытие.

Замечание. Особенность логического условия покрытия всех строк таблицы заключается в отсутствии отрицаний переменных. И, как следствие этого, приведение ее к минимальной ДНФ сводится к раскрытию скобок по свойству дистрибутивности и в применении свойств поглощения и идемпотентности.

Пример. Дана двоичная таблица покрытий размером 6×6.

Первая строка покрывается столбцом y_1 или столбцом y_6. Логическое условие покрытия первой строки записывается как $(y_1 \vee y_6)$. Логическое условие покрытия второй строки – $(y_2 \vee y_3)$, третьей строки – $(y_1 \vee y_4)$, четвертой строки – $(y_5 \vee y_6)$, пятой – $(y_1 \vee y_2)$ и последней – $(y_3 \vee y_4 \vee y_5)$. Логическое условие покрытия всех строк таблицы является

	y_1	y_2	y_3	y_4	y_5	y_6
z_1	1					1
z_2		1	1			
z_3	1			1		
z_4					1	1
z_5	1	1				
z_6			1	1	1	
c	40	30	40	50	50	30

конъюнкцией логических условий покрытия каждой строки:
$(y_1 \vee y_6) \wedge (y_2 \vee y_3) \wedge (y_1 \vee y_4) \wedge (y_5 \vee y_6) \wedge (y_1 \vee y_2) \wedge (y_3 \vee y_4 \vee y_5)$.

Многократное применение свойства дистрибутивности освобождает это выражение от скобок, а для устранения избыточности в ДНФ используется свойство поглощения:

$$(y_1 \vee y_6) \wedge (y_2 \vee y_3) \wedge (y_1 \vee y_4) \wedge (y_5 \vee y_6) \wedge (y_1 \vee y_2) \wedge (y_3 \vee y_4 \vee y_5)=$$
$$=((y_1 \vee y_6) \wedge (y_1 \vee y_4)) \wedge ((y_2 \vee y_3) \wedge (y_1 \vee y_2)) \wedge ((y_5 \vee y_6) \wedge (y_3 \vee y_4 \vee y_5))=$$
$$=(y_1 \vee y_4 \wedge y_6) \wedge (y_2 \vee y_1 \wedge y_3) \wedge (y_5 \vee y_6 \wedge (y_3 \vee y_4))=$$
$$=(y_1 \wedge y_2 \vee y_1 \wedge y_3 \vee y_2 \wedge y_4 \wedge y_6 \vee y_1 \wedge y_3 \wedge y_4 \wedge y_6)(y_5 \vee y_3 \wedge y_6 \vee y_4 \wedge y_6)=$$
$$=(y_1 \wedge y_2 \vee y_1 \wedge y_3 \vee y_2 \wedge y_4 \wedge y_6) \wedge (y_5 \vee y_3 \wedge y_6 \vee y_4 \wedge y_6)=$$
$$=y_1 \wedge y_2 \wedge y_5 \vee y_1 \wedge y_3 \wedge y_5 \vee y_2 \wedge y_4 \wedge y_5 \wedge y_6 \vee y_1 \wedge y_2 \wedge y_3 \wedge y_6 \vee y_1 \wedge y_3 \wedge y_6 \vee$$
$$\vee y_2 \wedge y_3 y_4 \wedge y_6 \vee y_1 \wedge y_2 \wedge y_4 \wedge y_6 \vee y_1 \wedge y_3 \wedge y_4 \wedge y_6 \vee y_2 \wedge y_4 \wedge y_6=$$
$$=y_1 \wedge y_2 \wedge y_5 \vee y_1 \wedge y_3 \wedge y_5 \vee y_1 \wedge y_3 \wedge y_6 \vee y_2 \wedge y_4 \wedge y_6.$$

Некоторые импликанты в вычислениях удаляются в результате применения к ним свойства поглощения. Упростить полученную ДНФ нельзя и, следовательно, форма – минимальна. По полученной минимальной ДНФ можно сделать следующий вывод:

- заданная двоичная таблица допускает четыре покрытия:
  $\pi_1=\{y_1, y_2, y_5\}$;
  $\pi_2=\{y_1, y_3, y_5\}$;
  $\pi_3=\{y_1, y_3, y_6\}$;
  $\pi_4=\{y_2, y_4, y_6\}$;
- стоимость найденных покрытий равна:
  $C(\pi_1)=c(y_1)+c(y_2)+c(y_5)=40+30+50=120$;
  $C(\pi_2)=c(y_1)+c(y_3)+c(y_5)=40+40+50=130$;
  $C(\pi_3)=c(y_1)+c(y_3)+c(y_6)=40+40+30=110$;
  $C(\pi_4)=c(y_2)+c(y_4)+c(y_6)=30+50+30=110$;
- исходная таблица допускает два покрытия минимальной стоимости: $\pi_3=\{y_1, y_3, y_6\}$ и $\pi_4=\{y_2, y_4, y_6\}$.

• Задачи и упражнения

① Формализацией каких из нижеприведенных суждений является высказывательная функция $\overline{x}_1 \to x_2 \vee x_3$:

– неверно, что x_1 является достаточным условием для x_2 и x_3;
– не x_1 является необходимым условием x_2 или x_3;
– x_2 или x_3, если только не x_1;
– x_2 или x_3, если не x_1;
– x_2 или x_3 является необходимым условием для не x_1.

② Даны три простых высказывания: A – «ясная погода»; B– «ветреная погода»; C– «дождливая погода». Формализовать следующее логическое суждение: «*Неверно, что погода будет безветренная либо пасмурная, если только не будет дождя*».

③ Записать в символическом виде и вычислить, какие из трех датчиков x_1, x_2 и x_3 исправны, а какие работают со сбоями, если известно, что:

– неверно, что если x_1 исправен, то и x_2 работает без сбоев;
– датчик x_1 сбоит, если сбоят датчики x_2 или x_3.

④ Проверить равносильность логических функций f и g:

– $f = \overline{A} \wedge (B \to C) \vee C \wedge (A \to B)$, $g = (A \to B \wedge C) \wedge (B \to B \wedge C)$;
– $f = \overline{(B \to A) \wedge (C \to \overline{B})} \vee \overline{A} \wedge \overline{C}$, $g = A \wedge C \to B \wedge C$;
– $f = \overline{\overline{B} \wedge (\overline{A} \to \overline{C})} \to C \wedge (B \to A)$, $g = C \wedge (B \to A)$.

⑤ Для заданных логических функций построить (найти) СовДНФ и СовКНФ:

– $F(x_1, x_2, x_3) = x_1 \wedge x_2 \wedge x_3 \vee \overline{x}_1 \wedge x_2 \wedge \overline{x}_3$;
– $F(a, b, c) = \overline{a \oplus \overline{b}} + \overline{c}$;
– $F(a, b, c) = \overline{a} \wedge \overline{b} \leftrightarrow \overline{c}$;
– $F(a, b, c) = a \oplus \overline{b} \vee \overline{c}$.

⑥ Для заданной таблицы найти все покрытия строк столбцами.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
p_1		1	1				1	
p_2	1		1	1			1	1
p_3	1				1	1		
p_4		1		1	1	1	1	1
p_5		1				1		1
p_6				1				

4. МИНИМИЗАЦИЯ (УПРОЩЕНИЕ) ЛОГИЧЕСКИХ ФУНКЦИЙ В КЛАССЕ ДНФ

4.1. Задача минимизации логических функций в классе ДНФ

Пусть $f(x_1, x_2, \ldots, x_n)$ – некоторая полностью определенная логическая функция, представленная в дизъюнктивно нормальной форме (ДНФ).

Под *сложностью функции* в ДНФ – $L(f(x_1, x_2, \ldots, x_n))$ понимают число первичных термов в ДНФ. Логическая функция в общем случае допускает множество представлений в ДНФ, которые могут иметь различные сложности. Для решения многих прикладных задач часто требуется найти такую ДНФ для заданной функции $f(x_1, x_2, \ldots, x_n)$, которая имеет минимальную сложность.

Пусть $F[f(x_1, x_2, \ldots, x_n)]$ – множество всех логических функций, эквивалентных $f(x_1, x_2, \ldots, x_n)$, представленных в ДНФ.

Функция $f^*(x_1, x_2, \ldots, x_n)$, $f^*(x_1, x_2, \ldots, x_n) \in F[f(x_1, x_2, \ldots, x_n)]$ называется минимальной ДНФ (МинДНФ) функции $f(x_1, x_2, \ldots, x_n)$, если для любой функции $f'(x_1, x_2, \ldots, x_n) \in F[f(x_1, x_2, \ldots, x_n)]$ справедливо, что $L(f^*(x_1, x_2, \ldots, x_n)) \leq L(f'(x_1, x_2, \ldots, x_n))$.

<u>Постановка задачи нахождения минимальной ДНФ</u>.

Дано:

$f(x_1, x_2, \ldots, x_n)$ – некоторая логическая функция, отличная от тождественного нуля.

Определить:

ДНФ $f^*(x_1, x_2, \ldots, x_n), f^*(x_1, x_2, \ldots, x_n) \in F[f(x_1, x_2, \ldots, x_n)]$, которая обладает минимальной сложностью.

Существуют разновидности этой задачи. В частности, иногда требуется найти все минимальные ДНФ для заданной функции $f(x_1, x_2, \ldots, x_n)$.

Из определения МинДНФ непосредственно следует и схема решения задачи:

1) для исходной функции $f(x_1, x_2, \ldots, x_n)$ найти все возможные ее представления в ДНФ (вычислить множество $F[f(x_1, x_2, \ldots, x_n)]$);

2) для каждой функции из $F[f(x_1, x_2, \ldots, x_n)]$ вычислить ее сложность;

3) из множества функций $F[f(x_1, x_2, \ldots, x_n)]$) выбрать ту форму (или те формы), которая имеет минимальную сложность.

Наиболее трудоемкий этап в этой схеме первый – поиск всех ДНФ для заданной функции.

Как отмечалось, ДНФ произвольной функции – это ее представление в виде дизъюнкции некоторого числа импликант:

$$f(x_1, x_2, \ldots, x_n)_{\text{ДНФ}} = c_1 \vee c_2 \vee \ldots \vee c_s = \bigvee_{j=1}^{s} c_j,$$

где c_j – некоторая импликанта ($1 \leq j \leq s$).

Тогда сложность $L(f(x_1, x_2, \ldots, x_n)) = \sum_{j=1}^{s} L(c_j)$

определяется числом импликант (s) и сложностью каждой импликанты в отдельности. При этом под сложностью импликанты понимается число первичных термов в ее представлении. Для поиска ДНФ, обладающей минимальной сложностью, необходимо уметь находить импликанты, которые обладают минимальной сложностью. Поиск таких импликант основан на свойствах эквивалентных преобразований алгебры логики, которые не выводят функцию из класса ДНФ, с одной стороны, а с другой – позволяют снизить сложность импликант.

Таким требованиям удовлетворяют свойства идемпотентности, поглощения и склеивания.

Минимальные по сложности импликанты функции $f(x_1, x_2, \ldots, x_n)$ называют простыми.

Представление логической функции в виде дизъюнкции всех простых импликант называют *сокращенной ДНФ* (СокДНФ).

Среди всех ДНФ заданной функции выделяют так называемые *тупиковые ДНФ* (ТДНФ). По определению, ТДНФ – это ДНФ, из которой нельзя удалить ни одной импликанты и ни одного терма без нарушения свойства эквивалентности исходной функции.

Лемма. Любая ТДНФ есть дизъюнкция некоторого числа простых импликант.

Действительно, если некоторая ДНФ содержит импликанту, которая не является простой, ее можно упростить за счет удаления избыточных термов применением правил алгебры логики к импликантам, которые не являются простыми. И, как следствие, такая ДНФ не принадлежит множеству ТДНФ.

Лемма позволяет дать новое определение МинДНФ.

> ***Минимальная ДНФ*** – это тупиковая ДНФ, обладающая минимальной сложностью.

Необходимо отметить, что в общем случае функция может иметь несколько минимальных ДНФ.

Проведенный анализ задачи позволяет модифицировать схему поиска минимальных ДНФ:

- найти все простые импликанты и построить сокращенную ДНФ;
- найти все тупиковые ДНФ;
- вычислить сложность тупиковых ДНФ и определить те из них, которые имеют минимальную сложность (МинДНФ).

Однако и эта схема не дает ответа на вопрос: в каком порядке и к каким импликантам в исходной функции следует применять свойство склеивания для получения всех ТДНФ?

4.2. Геометрическая интерпретация логических функций

Различные способы представления логических функций позволяют свести задачу минимизации логических функций от аналитических преобразований к работе с фиксированными структурами, допускающими алгоритмическое описание процедур. Существует два основных способа графического представления логических функций:

– задание функции с помощью гиперкуба;

– задание функции с помощью карты Карно.

Назначение этих способов – снизить трудоемкость нахождения простых импликант заданной функции. Оба эти представления позволяют при относительно небольших размерностях функции находить минимальную ДНФ.

4.2.1. Гиперкуб

Гиперкуб размерности n ($n = 0, 1, 2, 3, \ldots$) – это графический объект, который удовлетворяет следующим требованиям:

1) гиперкуб размерности n содержит 2^n вершин, каждой из которых присвоен уникальный двоичный код длины n;

2) вершины гиперкуба размещены по $n+1$ ярусам, каждому из которых сопоставлен номер j яруса ($j = 0, 1, \ldots, n$);

3) на j ярус ($j = 0, 1, \ldots, n$) выносятся те вершины гиперкуба, в коде которых содержится ровно j единиц;

4) вершины, лежащие на соседних ярусах, соединяются ребром, если соответствующие им коды отличны друг от друга ровно в одном разряде.

Примеры гиперкубов размерности n = 0, 1, 2, 3 приведены на рис. 4.1.

Каждая вершина гиперкуба (ее код) соответствует некоторой конституенте. Так, вершине с кодом 001 соответствует конституента

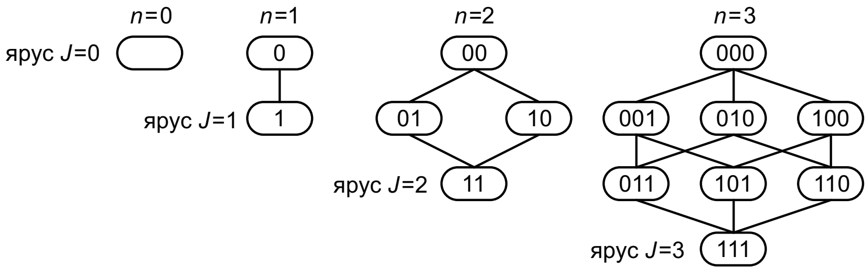


Рис. 4.1. Примеры гиперкубов размерности 0, 1, 2 и 3

$\bar{x}_1 \wedge \bar{x}_2 \wedge x_3$, а вершине с кодом 101 – конституента $x_1 \wedge \bar{x}_2 \wedge x_3$. Ребро (линия), которое соединяет две вершины, свидетельствует о том, что к этим вершинам, точнее, к соответствующим им конституентам, применимо правило склеивания: $\bar{x}_1 \wedge \bar{x}_2 \wedge x_3 \vee x_1 \wedge \bar{x}_2 \wedge x_3 = \bar{x}_2 \wedge x_3$, что приводит к снижению сложности в данном примере с 6 до 2. Таким образом, ребра гиперкуба потенциально определяют возможность тотального применения правила склеивания. Очевидно, что если вершины не соединены ребром, то правило склеивания к ним не применимо.

Подмножество вершин гиперкуба, образующее гиперкуб меньшей размерности, называют *интервалом*. Для обозначения интервалов используются двоичные коды, дополненные прочерками. Они ставятся в тех позициях, по которым происходит «склеивание». Так, вершины 001 и 101 образуют интервал размерности 1 с кодом «–01», а вершины 000, 100, 001, 101 образуют интервал размерности 2, код которого «–0–». Число прочерков в коде интервала определяет его размерность, ранг (r). Соответственно число вершин в интервале размерности r равно 2^r.

Каждой вершине гиперкуба (интервалу размерности 0), как отмечалось выше, соответствует конституента. Каждому интервалу, размерность которого больше 0, соответствует импликанта. Так, интервалу «–01» соответствует импликанта $\bar{x}_2 \wedge x_3$, а интервалу «–0–» – импликанта $\bar{x}_2$.

Замечание 1. Предлагается убедиться, что гиперкуб размерности 3 содержит 8 интервалов размерности 0, 12 интервалов размерности 1 и 6 интервалов размерности 2.

Замечание 2. Использование для обозначения переменных символа x с соответствующими индексами не является существенным. Можно использовать и другие символы, такие как y, z, w и т.п.

Для задания логической функции $f(x_1, x_2, \ldots, x_n)$ на гиперкубе необходимо указать ту часть гиперкуба размерности n, в точках которого функция принимает значение 1.

Пример. Дана логическая функция $f(x_1, x_2, x_3, x_4)_{\text{СовДНФ}} = \bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge \bar{x}_4 \vee \bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge x_4 \vee x_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge \bar{x}_4 \vee \bar{x}_1 \wedge \bar{x}_2 \wedge x_3 \wedge x_4 \vee x_1 \wedge x_2 \wedge \bar{x}_3 \wedge \bar{x}_4 \vee \bar{x}_1 \wedge x_2 \wedge x_3 \wedge x_4 \vee x_1 \wedge x_2 \wedge \bar{x}_3 \wedge x_4 \vee x_1 \wedge x_2 \wedge x_3 \wedge \bar{x}_4 \vee x_1 \wedge x_2 \wedge x_3 \wedge x_4$. На рис. 4.2 приведено ее представление на гиперкубе.

Связи между вершинами, как отмечалось, определяют все возможные способы применения правила склеивания для понижения сложности ДНФ. Функция, представленная на рис. 4.2, содержит:

- 9 интервалов размерности 0;
- 10 интервалов размерности 1;
- 1 интервал размерности 2.

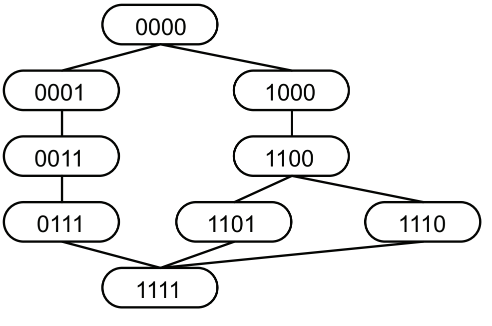


Рис. 4.2. Пример задания функции на гиперкубе

4.2.2. Карты Карно

Карты Карно (диаграммы Вейча) реализуют графоаналитический метод представления и упрощения логических функций.

Карта Карно – это двумерная таблица специального вида, имеющая 2^n клеток, где n – число логических переменных. На рис. 4.3 приведены примеры карт Карно размерности $n = 0, 1, 2, 3, 4, 5, 6$.

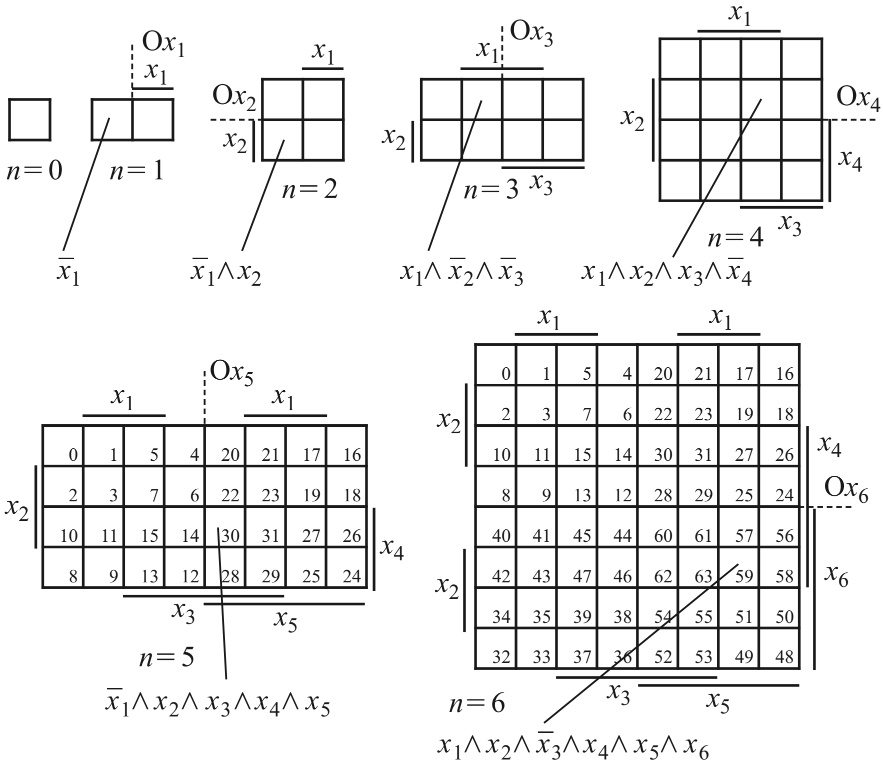


Рис. 4.3. Карты Карно для $n = 0, 1, 2, 3, 4, 5, 6$

Для карты Карно определено разбиение множества логических переменных $\{x_1, x_2, \ldots, x_n\}$ на два непересекающихся подмножества переменных. Первое из них (Mcol) связано со столбцами, а второе (Mrow) связано со строками. Для такого разбиения справедливо, что

$\mathrm{Mcol} \cap \mathrm{Mrow} = \varnothing$ и $\mathrm{Mcol} \cup \mathrm{Mrow} = \{x_1, x_2, \ldots, x_n\}$. Так, на рис. 4.3 для карты Карно с $n=5$ $\mathrm{Mrow} = \{x_2, x_4\}$ и $\mathrm{Mcol} = \{x_1, x_3, x_5\}$, а для карты с $n=6$ $\mathrm{Mrow} = \{x_2, x_4, x_6\}$ и $\mathrm{Mcol} = \{x_1, x_3, x_5\}$. В общем случае $|\mathrm{Mrow}| = n \operatorname{div} 2$, а $|\mathrm{Mcol}| = (n+1) \operatorname{div} 2$, где div – операция целочисленного деления. Число строк карты определяется как $2^{|\mathrm{Mrow}|}$, число столбцов равно $2^{|\mathrm{Mcol}|}$.

Для связывания клеток карты с первичными термами переменных вдоль границ таблицы указываются *линейные* метки термов различных переменных $x_i^1 = x_i$ $(1 \leq i \leq n)$. Переменные термов в линейных метках столбцов выбираются из Mcol, а переменные термов линейных меток строк выбираются из Mrow. Эти линейные метки выделяют группу столбцов (группу строк), которые называют *прямой зоной* карты Карно по переменной x_i. Все остальные столбцы (строки), не попавшие в прямую зону по переменной x_i, по умолчанию имеют линейную метку $x_i^0 = \overline{x}_i$ и образуют инверсную зону карты по переменной x_i. Линию, которая разделяет прямую и инверсную зоны по переменной x_i, называют *осью по переменной* x_i и обозначают как $\mathrm{O}x_i$. В общем случае карта Карно допускает несколько осей по одной и той же переменной.

Любая клетка карты Карно одновременно находится в n зонах по каждой из n переменных. Конъюнкция первичных термов этих зон образует конституенту, соответствующую выбранной клетке. На рис. 4.3 приведены примеры конституент для различных клеток карт Карно. Часто клетки нумеруются натуральными числами, двоичный код которых однозначно определяют соответствующие конституенты.

Важное свойство карт Карно заключается в способе размещения переменных (линейных меток) в карте Карно. При переходе от каждой клетки к соседней изменяется первичный терм только одной переменной в конституенте. Иными словами, две соседние клетки всегда находятся в разнотипных зонах относительно некоторой переменной. И следовательно, к соответствующим конституентам может быть применено правило склеивания по соответствующей переменной.

Успех применения карт Карно, особенно при больших значениях *n*, в значительной степени определяется используемой *схемой развертывания*, лежащей в основе построения карт большей размерности. Под схемой развертывания карт Карно понимается алгоритм, который позволяет по карте размерности *n* (исходной карте) строить карты размерности *n*+1, *n*+2 и т.д. При этом для уменьшения трудоемкости построения и обработки карты необходимо максимально использовать данные из таблицы размерности *n*, такие как нумерация клеток и размещение линейных меток.

Существует несколько схем развертывания карт Карно.

Увеличение размерности *n* карты Карно на *n*+1 происходит по следующему алгоритму:

- выбирается новая ось, совпадающая с одной из сторон исходной карты;
- исходная карта зеркально отображается относительно выбранной оси;
- производится нумерация новой части карты таким образом, чтобы нумерация старой части осталась без изменений.

При этом новая часть карты совпадает с новой прямой зоной, а исходная карта Карно – с инверсной новой зоной. Возможные схемы развертывания (увеличение размерности) карт Карно показаны на рис. 4.4.

Карты Карно, соответствующие схемам развертывания, показанным на рис. 4.4, *а*, *б*, представляют собой таблицы-строки $1\times2n$ либо таблицы-столбцы размерности $2n\times1$. При числе переменных $n>4$ они крайне неудобны в работе. Карты Карно по схемам развертывания, показанным на рис. 4.4, *в*, *г*, напротив, являются таблицами с размерностью $2^{n/2}\times2^{n/2}$ (при четном числе переменных) или с размерностью $2^{(n+1)/2}\times2^{(n+1)/2-1}$ (при нечетном числе переменных) и при любом числе переменных удобны в работе. Схема, показанная на рис. 4.4, *в*, представляет собой разворачивающуюся спираль, а схема на рис. 4.4, *г* – «ступеньки».

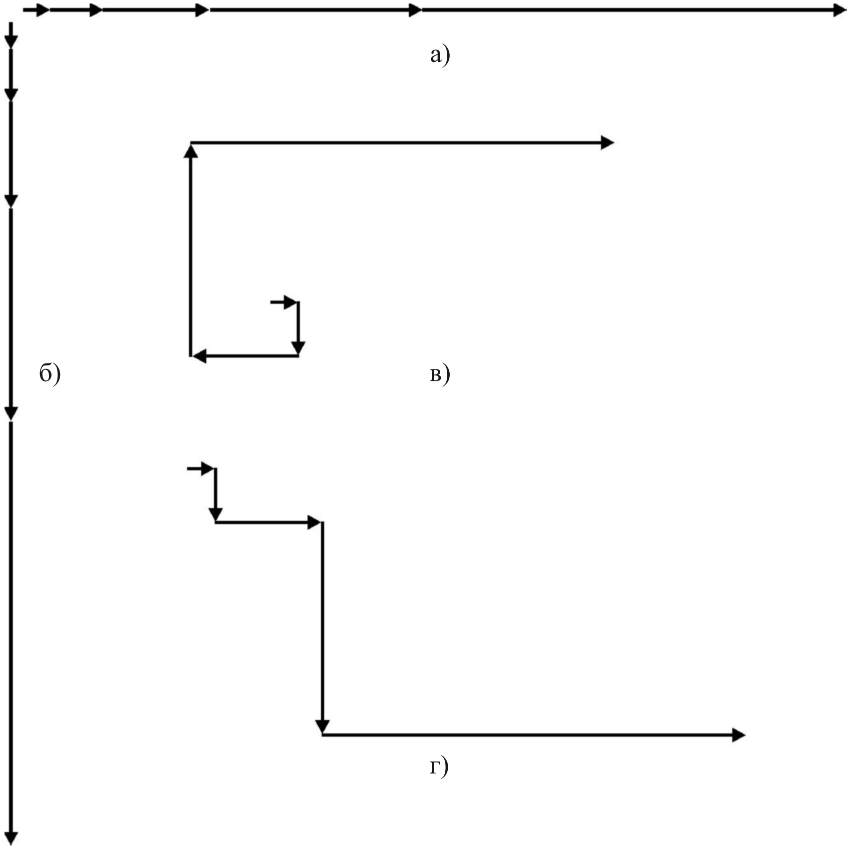


Рис. 4.4. Схемы развертывания карт Карно

При ступенчатой схеме развертывания карт Карно выбирается новая ось, совпадающая либо с большей нижней стороной карты Карно, если исходная карта прямоугольная, либо с правой стороной карты, если карта – квадрат.

Для снижения трудоемкости построения карт Карно большой размерности применяют *ступенчатую схему развертывания* (схему перехода от карты размерности n к карте размерности $n+1$).

На каждом шаге развертывания карты Карно по *ступенчатой схеме* справедливо неравенство $0 \le |\mathrm{Mcol}| - |\mathrm{Mrow}| \le 1$. Это означает, что любая карта Карно либо квадратная ($|\mathrm{Mcol}| - |\mathrm{Mrow}| = 0$), либо прямоугольная ($|\mathrm{Mcol}| - |\mathrm{Mrow}| = 1$).

Операция развертывания (увеличения размерности с n на n+1) карты Карно заключается в следующем:

– Если исходная таблица квадратная, выбирается новая ось, совпадающая с правой стороной карты, и исходная таблица зеркально отображается относительно этой оси вместе с линейными метками. К достроенным столбцам снизу присоединяется новая линейная метка, соответствующая переменной x_{n+1}.

– Если исходная таблица прямоугольная, выбирается новая ось, совпадающая с нижней (большей) стороной карты, и исходная таблица зеркально отображается относительно этой оси вместе с линейными метками. К достроенным строкам справа присоединяется новая линейная метка, соответствующая переменной x_{n+1}.

– При построении зеркального отображения номера новых клеток увеличиваются на 2^n.

По рис. 4.3 можно проследить ступенчатую схему развертывания, анализируя последовательность карт с увеличивающейся размерностью.

Для карт Карно введено понятие *интервала размерности* (*ранга*) r. Интервал размерности r определяется как множество, состоящее из 2^r клеток, которые расположены симметрично относительно фиксированного набора из r осей различных переменных.

Интервал размерности 0 содержит одну клетку карты.

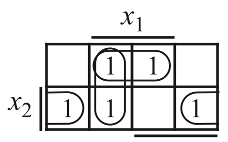
Две клетки образуют интервал размерности 1, если клетки симметричны некоторой оси.

Два интервала размерности 1 образуют интервал размерности 2, если исходные интервалы симметричны относительно некоторой оси. Четыре клетки, которые входят в этот интервал, симметричны относительно двух осей.

Соответственно, по индукции два интервала размерности 2 образуют интервал размерности 3, если исходные интервалы симметричны относительно некоторой оси. Восемь клеток, входящие в этот интервал, симметричны относительно трех осей.

Каждому интервалу размерности r ставится в соответствие импликанта – конъюнкция n-r первичных термов, сохраняющих свое значение для всех его клеток.

На рис. 4.5 приведены примеры карт Карно с выделенными интервалами размерности 1 (рис. 4.5, *а*) и размерности 2 (рис. 4.5, *б*).



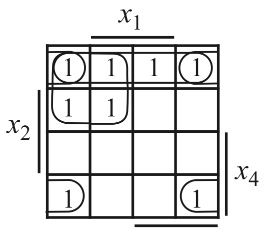


Рис. 4.5. Примеры выделения интервалов

Выделенным на рис. 4.5, *а* интервалам размерности 1 соответствуют импликанты $x_1 \wedge \overline{x}_2$, $x_1 \wedge \overline{x}_3$ и $\overline{x}_1 \wedge x_2$. А выделенным на рис. 4.5, *б* интервалам размерности 2 соответствуют импликанты $\overline{x}_3 \wedge \overline{x}_4$, $\overline{x}_1 \wedge \overline{x}_2$ и $\overline{x}_2 \wedge \overline{x}_4$.

Сравнивая гиперкуб размерности n с картой Карно той же размерности, можно отметить определенное сходство между ними. Вершина гиперкуба отождествляется с клеткой карты Карно. И вершине, и клетке соответствует определенная конституента. Ребро гиперкуба отождествляется с отношением соседства между клетками, обладающими способностью порождать интервалы большей размерности. И карта Карно, и гиперкуб являются носителями всех возможных последовательностей применения правила склеивания конституент, связанных как с вершинами гиперкуба, так и с клетками карты.

Для задания логической функции $f(x_1, x_2, \ldots, x_n)$ на карте Карно необходимо расставить единицы в тех клетках таблицы, которые соответствуют конституентам в ее совершенной ДНФ. На рис. 4.6 приведен пример задания логической функции четырех переменных на карте Карно. Это задание можно сравнить с ее заданием на гиперкубе (рис. 4.2).

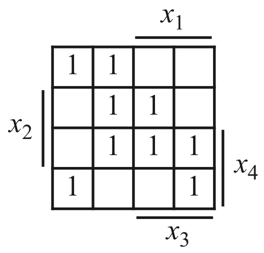


Рис. 4.6. Задание функции на карте Карно

Основным достоинством карт Карно является то, что с их помощью можно быстро и просто находить некоторую минимальную ДНФ.

4.3. Метод Квайна–Мак-Класки

Метод позволяет найти все тупиковые и все минимальные ДНФ для заданной логической функции. Он определяется схемой, которая представлена на рис. 4.7.

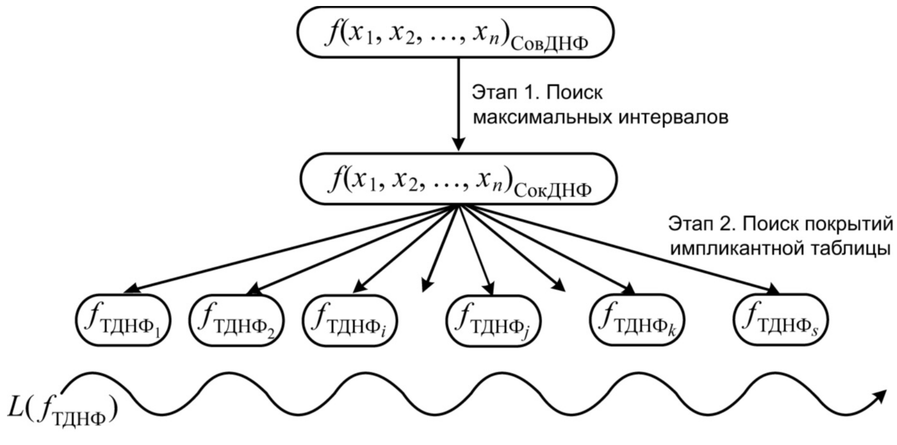


Рис. 4.7. Схема метода Квайна–Мак-Класки

Предполагается, что исходная функция задана в СовДНФ (или перечислением точек гиперкуба, в которых исходная функция принимает значение 1). Метод позволяет заменить операции эквивалентных преобразований, обеспечивающие нахождение простых импликант, тупиковых и минимальных ДНФ на операции манипулирования с интервалами размерности 0, 1, 2 и т.д. для поиска максимальных интервалов и покрытий импликантной таблицы. Метод включает два последовательных этапа: этап 1 и этап 2. Результатом первого этапа является список максимальных интервалов и, соответственно, сокращенная ДНФ заданной функции. Результат второго этапа – список тупиковых ДНФ с указанием, какие из них являются минимальными.

Этап 1. Поиск максимальных интервалов.

Этап включает начальный шаг и итерационный шаг, число выполнений которого зависит от исходной функции.

Начальный шаг. Нахождение интервалов размерности $r=0$.

Интервалам размерности 0 соответствуют гиперкубы размерности 0. И следовательно, все интервалы размерности 0 – это точки гиперкуба, в которых функция принимает значение 1. Иными словами, все вершины функции при ее задании на гиперкубе на начальном шаге объявляются максимальными.

Итерационный шаг. Нахождение интервалов размерности r ($r=1, 2, \ldots$).

Исходными данными для этого шага являются интервалы размерности $r-1$, найденные на предыдущем шаге. Число прочерков в коде интервала определяет его размерность. Все интервалы размерности $r-1$ выписываются в столбик по ярусам $j=0$, 1, 2 и т.д. таким образом, что на j-й ярус выносятся те интервалы, в коде которых содержится ровно j единиц. Производятся все попарные сравнения интервалов, лежащих на соседних ярусах. Если коды сравниваемых интервалов различаются ровно в одном разряде, то они порождают интервал размерности r. Код интервала отличается от кода порождающих его интервалов значением различающего их разряда: в этом разряде ставится символ «–» прочерка. Интервалы размерности r–1, порождающие интервал размерности r, максимальными не являются. Все интервалы, которые вошли в интервалы большей размерности и, следовательно, не являются максимальными, помечаются специальным символом «✓».

Правило остановки. Выполнение итерационного шага прекращается, если в результате его выполнения не были найдены интервалы большей размерности. При этом максимальными объявляются те интервалы, которые не отмечены символом «✓».

Этап 2. Поиск покрытий импликантной таблицы.

На этом этапе составляется таблица покрытий. Каждой строке ставится в соответствие двоичная точка, в которой функция принимает значение 1. Каждому столбцу ставится в соответствие максимальный интервал, найденный на первом этапе. Если точка, соответствующая i-й строке, входит в j-й интервал, на пересечении i-й строки и j-го столбца ставится 1. Заполненную таким образом таблицу называют *импликантной таблицей*.

Решение задачи покрытия строк столбцами позволяет найти все тупиковые ДНФ. Каждому покрытию импликантной таблицы соответствует набор максимальных интервалов, а дизьюнкция соответствующих им простых импликант – тупиковая ДНФ (ТДНФ).

Рассмотрим применение этого метода для решения задачи: найти все тупиковые и минимальные ДНФ для функции $f(x_1, x_2, x_3, x_4)=$ $=\overline{x}_1\wedge\overline{x}_2\wedge\overline{x}_3\wedge\overline{x}_4\vee\overline{x}_1\wedge\overline{x}_2\wedge\overline{x}_3\wedge x_4\vee x_1\wedge\overline{x}_2\wedge\overline{x}_3\wedge\overline{x}_4\vee\overline{x}_1\wedge x_2\wedge$ $\wedge\overline{x}_3\wedge x_4\vee x_1\wedge x_2\wedge\overline{x}_3\wedge\overline{x}_4\vee x_1\wedge x_2\wedge\overline{x}_3\wedge x_4\vee x_1\wedge x_2\wedge x_3\wedge\overline{x}_4\vee x_1\wedge x_2\wedge x_3\wedge x_4$.

Эта функция представлена в виде СовДНФ. Ее сложность $L(f(x_1, x_2, x_3, x_4))=32$.

Решение задачи.

Этап 1 (табл. 4.1).

Таблица 4.1

Выделение максимальных интервалов

r=0	r=1	r=2	Комментарии
✓ 0000	000–		При сравнении интервалов, лежащих на соседних ярусах, интервал большей размерности образуют те из них, которые различаются ровно в одном разряде. В этом разряде появляется прочерк в интервале большей размерности. Все интервалы, не отмеченные символом «✓», являются максимальными. Если получается повторение интервалов, достаточно выписать его один раз.
✓ 0001	–000		
✓ 1000	0–01		
✓ 0101	1–00		
✓ 1100	✓ 110–		
✓ 1101	–101		
✓ 1110	✓ 11–0	11––	
✓ 1111	✓ 11–1		
	✓ 111–		

В результате выполнения первого этапа получено шесть максимальных интервалов, пять из которых имеют размерность 1, а один – размерность 2. Список максимальных интервалов I_{max}={000–, –000, 0–01, 1–00, –101, 11––}. Каждому максимальному интервалу, как отмечалось выше, ставится в соответствие простая импликанта. В результате сокращенная ДНФ для заданной функции:

$$f(x_1,\ x_2,\ x_3,\ x_4)_{\text{СокДНФ}} = \overline{x}_1\wedge\overline{x}_2\wedge\overline{x}_3\vee\overline{x}_2\wedge\overline{x}_3\wedge\overline{x}_4\vee\overline{x}_1\wedge\overline{x}_3\wedge x_4\vee x_1\wedge$$
$$\wedge\overline{x}_3\wedge\overline{x}_4\vee x_2\wedge\overline{x}_3\wedge x_4+x_1\wedge x_2.$$

Сложность $L(f(x_1,\ x_2,\ x_3,\ x_4)_{\text{СокДНФ}})=17$.

<u>Этап 2</u> (табл. 4.2).

Поиск покрытия импликантной таблицы сводится к задаче нахождения покрытий минимальной стоимости, в которой стоимости столбцов равны между собой.

Каждой строке соответствует точка, в которой функция принимает значение 1, а каждому столбцу – максимальный интервал. На пересечении строки и столбца ставится 1, если соответствующая строке точка входит в интервал, соответствующий столбцу. Для поиска всех

Таблица 4.2

Импликантная таблица

I_{max} / точки	a 000–	b –000	c 0–01	d 1–00	e –101	f 11––
0000	1	1				
0001	1		1			
1000		1		1		
0101			1		1	
1100				1		1
1101					1	1
1110						1
1111						1

покрытий строк столбцами составлено логическое условие покрытия:

$$(a\vee b)\wedge(a\vee c)\wedge(b\vee d)\wedge(c\vee e)\wedge(d\vee f)\wedge(e\vee f)\wedge f\wedge f.$$

В результате раскрытия скобок по свойству дистрибутивности и применения свойств поглощения и идемпотентности это условие приводится к ДНФ:

$$(a\vee b)\wedge(a\vee c)\wedge(b\vee d)\wedge(c\vee e)\wedge(d\vee f)\wedge(e\vee f)\wedge f\wedge f=(b\vee a\wedge d)\wedge$$
$$\wedge(c\vee a\wedge e)\wedge f=b\wedge c\wedge f\vee a\wedge b\wedge e\wedge f\vee a\wedge d\wedge c\wedge f\vee a\wedge d\wedge e\wedge f.$$

Эту ДНФ нельзя упростить. И следовательно, таблица покрытий допускает четыре покрытия:

$$\pi_1=\{b,\ c,\ f\};\ \pi_2=\{a,b,e,f\};\ \pi_3=\{a,d,c,f\};\ \pi_4=\{a,d,e,f\}.$$

Каждому покрытию соответствует тупиковая ДНФ. Тупиковая ДНФ определяется как дизъюнкция простых импликант, которые образуют покрытие:

$$f_{\text{ТДНФ1}}=\bar{x}_2\wedge\bar{x}_3\wedge\bar{x}_4\vee\bar{x}_1\wedge\bar{x}_3\wedge x_4\vee x_1\wedge x_2,\qquad L(f_{\text{ТДНФ1}})=8;$$

$$f_{\text{ТДНФ2}}=\bar{x}_1\wedge\bar{x}_2\wedge\bar{x}_3\vee\bar{x}_2\wedge\bar{x}_3\wedge\bar{x}_4\vee x_2\wedge\bar{x}_3\wedge x_4\vee x_1\wedge x_2,\quad L(f_{\text{ТДНФ2}})=11;$$

$$f_{\text{ТДНФ3}}=\bar{x}_1\wedge\bar{x}_2\wedge\bar{x}_3\vee\bar{x}_2\wedge\bar{x}_3\wedge\bar{x}_4\vee\bar{x}_1\wedge\bar{x}_3\wedge x_4\vee x_1\wedge x_2,\quad L(f_{\text{ТДНФ3}})=11;$$

$$f_{\text{ТДНФ4}}=x_1\wedge\bar{x}_2\wedge\bar{x}_3\vee x_1\wedge\bar{x}_3\wedge\bar{x}_4\vee x_2\wedge\bar{x}_3\wedge x_4\vee x_1\wedge x_2,\quad L(f_{\text{ТДНФ4}})=11.$$

Анализ сложности полученных тупиковых ДНФ свидетельствует о том, что задача имеет одну минимальную ДНФ:

$f_{\text{МинДНФ}}=f_{\text{ТДНФ1}}=\overline{x}_2\wedge\overline{x}_3\wedge\overline{x}_4\vee\overline{x}_1\wedge\overline{x}_3\wedge x_4\vee x_1\wedge x_2$ и ее сложность $L(f_{\text{МинДНФ}})=8$.

На рис. 4.8 приведено представление заданной функции на гиперкубе с указанием максимальных интервалов, образующих одно из покрытий. Именно этому покрытию соответствует минимальная ДНФ.

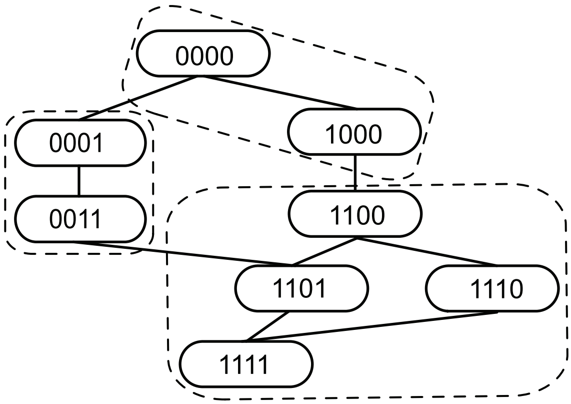


Рис. 4.8. Задание функции на гиперкубе
(пунктиром выделены максимальные интервалы, образующие покрытия)

Гиперкуб можно использовать для контроля за правильностью применения метода Квайна–Мак-Класки. Действительно, из примера видно, что функция включает 8 интервалов размерности 0, размещенных на 5 ярусах. Функция включает 9 интервалов размерности 1, из них – 5 являются максимальными: 000–, –000, 0–01, 1–00, –101. Четыре точки образуют интервал размерности 2 (11––). Выделенные на рис. 4.8 области соответствуют максимальным интервалам, которые образуют покрытие всех единичных точек. Именно это покрытие порождает МинДНФ. Все остальные покрытия можно продемонстрировать на гиперкубе подобным образом.

4.4. Метод минимизации по картам Карно

Поиск минимальной ДНФ по карте Карно состоит из двух этапов:

– Поиск минимального подмножества интервалов максимально высокого ранга, которые покрывают все единичные клетки таблицы. При этом одна и та же единичная клетка может входить в различные интервалы.

– Построение минимальной ДНФ посредством дизъюнкции всех импликант, каждая из которых соответствует найденному интервалу.

Сформулируем два правила, согласно которым каждому максимальному интервалу ранга r можно поставить в соответствие импликанту.

1. Если интервал (множество клеток, его образующих) пересекается осью Оx_i, т.е. часть клеток этого интервала лежит в зоне x_i, а часть – в зоне $\overline{x}_i$, то переменная x_i в соответствующую этому интервалу импликанту не войдет.

2. Если интервал целиком лежит в зоне x_i ($\overline{x}_i$), то в соответствующую этому интервалу импликанту войдет терм x_i ($\overline{x}_i$).

Каждой клетке карты Карно n переменных соответствует n соседних клеток. Все соседние клетки симметричны относительно какой-либо оси карты.

На рис. 4.9 для карты Карно четырех переменных показаны варианты соседних клеток, которые используются для отыскания максимальных интервалов.

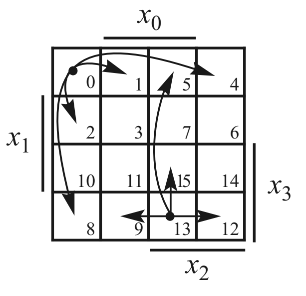


Рис. 4.9. Варианты соседних клеток для карты Карно при n=4

В качестве примера рассмотрим минимизацию логической функции, заданной с помощью карты Карно.

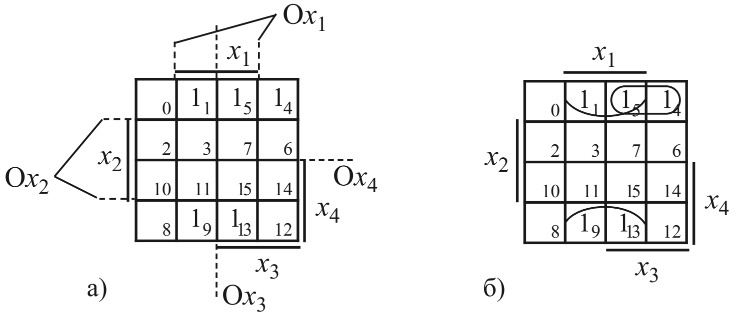


Рис. 4.10. Минимизация полностью определенной логической функции

В качестве исходного интервала 0-го ранга (рис. 4.10, *а*) выберем единичную клетку с номером 1. Интервал 1-го ранга можно получить, присоединяя к клетке 1 симметричную ей относительно оси Ox_2 единичную клетку с номером 5. Далее находим симметричный найденному интервалу 1-го ранга относительно оси Ox_3 интервал 1-го ранга, содержащий единичные клетки с номерами 9 и 13. Таким образом, четыре единичные клетки 1, 5, 9 и 13 образуют интервал 2-го ранга. Увеличить ранг этого интервала нельзя, так как в карте Карно отсутствуют четыре единичные клетки, симметричные ранее найденным (1, 5, 9, 13) относительно какой-либо оси карты. В найденный интервал не вошла единичная клетка с номером 4, которую примем за новый исходный интервал 0-го ранга. Увеличить ранг этого интервала можно, присоединяя к клетке 4 симметричную ей относительно оси Ox_0 единичную клетку с номером 5. Найденный интервал (4, 5) является максимальным, так как его ранг увеличить нельзя. Таким образом, все единичные клетки карты Карно для рассматриваемой функции «покрываются» двумя максимальными интервалами: 2-го ранга (1, 5, 9, 13) и 1-го ранга (4, 5). Оба интервала показаны на рис. 4.10, *б*. Первому интервалу соответствует импликанта $x_1 \wedge x_2$, а второму – импликанта $\overline{x}_2 \wedge x_3 \wedge \overline{x}_4$. Дизъюнкция этих импликант определяет минимальную ДНФ $\overline{x}_1 \wedge x_2 \vee \overline{x}_2 \wedge x_3 \wedge \overline{x}_4$.

• Задачи и упражнения

① Проверить, какие из следующих дизъюнктивно нормальных форм являются минимальными:

- $f_1(x_1, x_2, x_3)=\bar{x}_1\wedge\bar{x}_3\vee\bar{x}_1\wedge x_2\vee x_1\wedge x_3$;
- $f_2(x_1, x_2, x_3)=\bar{x}_1\wedge\bar{x}_2\vee\bar{x}_1\wedge x_2\vee x_2\wedge x_3$;
- $f_3(x_1, x_2, x_3)=\bar{x}_2\wedge\bar{x}_3\vee\bar{x}_1\wedge x_3\vee x_1\wedge x_2$;
- $f_4(x_1, x_2, x_3)=\bar{x}_1\wedge\bar{x}_2\wedge\bar{x}_3\vee\bar{x}_1\wedge x_3$;
- $f_5(x_1, x_2, x_3)=\bar{x}_1\wedge\bar{x}_2\wedge\bar{x}_3\vee x_1\wedge x_3$;
- $f_6(x_1, x_2, x_3)=\bar{x}_1\wedge\bar{x}_2\wedge\bar{x}_3\vee\bar{x}_1\wedge x_2\wedge x_3$;
- $f_7(x_1, x_2, x_3)=\bar{x}_1\wedge\bar{x}_2\wedge\bar{x}_3\vee\bar{x}_1\wedge x_2\wedge x_3\vee x_1\wedge x_2\wedge\bar{x}_3$.

② Для следующих функций, в которых единичные конституенты заданы соответствующими числовыми эквивалентами, найти все тупиковые и минимальные дизъюнктивные нормальные формы. Задать эти функции на гиперкубе и найти все максимальные интервалы:

- $g_1(x_1, x_2, x_3, x_4)=\vee(2, 3, 4, 6, 7, 8, 10, 11, 12, 15)$;
- $g_2(x_1, x_2, x_3, x_4)=\vee(0, 2, 3, 5, 7, 8, 9, 10, 11, 13)$;
- $g_3(x_1, x_2, x_3, x_4)=\vee(0, 2, 5, 6, 7, 8, 10, 12, 13, 14)$.

③ Для следующих функций, в которых единичные конституенты заданы соответствующими числовыми эквивалентами, найти любую минимальную дизъюнктивную нормальную форму. Задать эти функции на карте Карно и указать на ней покрытие, соответствующее найденной минимальной ДНФ:

- $h_1(x_1, x_2, x_3, x_4)=\vee(1, 2, 3, 5, 7, 8, 9, 10, 13, 15)$;
- $h_1(x_1, x_2, x_3, x_4, x_5)=\vee(0, 3, 5, 8, 10, 12, 14, 15, 17, 25, 27, 31)$;
- $h_1(x_1, x_2, x_3, x_4)=\vee(1, 3, 7, 9, 13, 15)$.

5. ФУНКЦИОНАЛЬНО ПОЛНЫЕ СИСТЕМЫ

5.1. Суперпозиция функций

Пусть $S=\{\varphi_1(x_1, x_2, \ldots, x_{n1}), \varphi_2(x_1, x_2, \ldots, x_{n1}), \ldots, \varphi_k(x_1, x_2, \ldots, x_{nk})\}$ – некоторая система функций, не обязательно логических. Говорят, что функция $f(x_1, x_2, \ldots, x_n)$ есть ***суперпозиция системы функций*** из S, если:

- функция $f(x_1, x_2, \ldots, x_n)$ получена из некоторой функции $\varphi_i(x_1, x_2, \ldots, x_{ni})$ из S путем переименования переменных в φ_i;
- функция $f(x_1, x_2, \ldots, x_n)$ получена из некоторой функции $\varphi_i(x_1, x_2, \ldots, x_{ni})$ из S путем подстановки вместо некоторых переменных в $\varphi_i(x_1, x_2, \ldots, x_{ni})$ функций из S;
- функция $f(x_1, x_2, \ldots, x_n)$ получена в результате конечного числа применений предыдущих двух правил.

Введенное определение является основой для построения исчислений. В частности, в математическом анализе операцию суперпозиции используют для замены (подстановки) переменных при определении сложных функций.

Пример. Пусть система S состоит из одной функции – функции сложения: $S=\{\varphi(x_1, x_2)=x_1 \vee x_2\}$. Вопрос: являются ли следующие три функции суперпозицией заданной системы S?

а) $f(x_1, x_2, x_3)=x_1 \vee (x_2 \vee x_3)$;

б) $g(x)=x$;

в) $h(x_1, x_2)=x_1 \vee \vee x_2$.

Первая функция является суперпозицией заданной, так как она представима в следующем виде:

$f(x_1, x_2, x_3) = x_1 \vee (x_2 \vee x_3) = \varphi(x_1, \varphi(x_2, x_3))$.

Вторая функция также есть суперпозиция системы S: $g(x) = \varphi(x, x)$.

Третья функция не является суперпозицией системы S.

Суперпозиция позволяет решать и обратную задачу: по заданной суперпозиции получить аналитическое (формульное) представление функции. Так, суперпозиция $\varphi(\varphi(x_1, x_2), \varphi(x_2, x_3))$, где $\varphi(x_1, x_2) = x_1 \vee x_2$, эквивалентна формуле $((x_1 \vee x_2) \vee (x_2 \vee x_3))$.

Пусть $S = \{\varphi_1(x_1, x_2, \ldots, x_{n1}), \varphi_2(x_1, x_2, \ldots, x_{n2}), \ldots, \varphi_k(x_1, x_2, \ldots, x_{nk})\}$ – некоторая система функций, не обязательно логических. Множество всех функций, которые являются суперпозицией системы S

$\Im\,[\varphi_1(x_1, x_2, \ldots, x_{n1}), \varphi_2(x_1, x_2, \ldots, x_{n2}), \ldots, \varphi_k(x_1, x_2, \ldots, x_{nk})]$,

называют *замыканием* S.

Пусть E_2 – совокупность всех логических функций. Система логических функций $S = \{\varphi_1(x_1, x_2, \ldots, x_{n1}), \varphi_2(x_1, x_2, \ldots, x_{n2}), \ldots, \varphi_k(x_1, x_2, \ldots, x_{nk})\}$ функционально полна, если любая функция из E_2 есть суперпозиция системы S.

Функционально полную систему функций называют логическим *базисом*, если удаление из S любой функции приводит к потере свойства функциональной полноты.

Учитывая, что ранее была определена форма представления логических функций в виде СовДНФ, для которой используется система из трех функций $S = \{\varphi_1(x_1, x_2) = x_1 \vee x_2,\ \varphi_2(x_1, x_2) = x_1 \wedge x_2,\ \varphi_3(x) = \overline{x}\}$, можно предположить, что эта система – функционально полна в E_2.

5.2. Классы логических функций

Среди всех логических функций можно выделить функции, суперпозиция которых сохраняет определенное свойство исходных функций. Множество таких функций называют классом (замыканием) относительно этого свойства. К таким классам относятся:

- K_0 – класс функций, сохраняющих константу 0;
- K_1 – класс функций, сохраняющих константу 1;
- $K_л$ – класс линейных функций;
- $K_м$ – класс монотонных функций;
- $K_с$ – класс самодвойственных функций.

Класс функций, сохраняющих константу 0 (K_0)

По определению, класс K_0 содержит все те функции, которые при нулевых значениях всех переменных принимают значение 0: $K_0=\{ f(x_1, x_2, \ldots, x_n) | f(0, 0, \ldots, 0)=0\}$. Так, функция $\varphi(x_1, x_2)=x_1 \wedge x_2$ принадлежит классу K_0, а функция $\psi(x)=\overline{x}$ не принадлежит классу K_0.

Класс функций, сохраняющих константу 1 (K_1)

По определению, класс K_1 содержит все те функции, которые при единичных значениях всех переменных принимают значение 1: $K_1=\{ f(x_1, x_2, \ldots, x_n) | f(1, 1, \ldots, 1)=1\}$. Так, функция $\varphi(x_1, x_2)=x_1 \wedge x_2$ принадлежит классу K_1, а функция $\psi(x)=\overline{x}$ не принадлежит классу K_1.

Класс линейных функций ($K_л$)

По определению, класс $K_л$ содержит все те функции, которые представимы в виде разложения в ряд по модулю 2:

$K_л=\{ f(x_1, x_2, \ldots, x_n) | f(x_1, x_2, \ldots, x_n)=c_0 \oplus c_1 \wedge x_1 \oplus c_2 \wedge x_2 \oplus \ldots \oplus c_n \wedge x_n\}$, где коэффициенты $c_0, c_1, c_2, \ldots, c_n$ при переменных принимают значение 0 либо 1. Например, функция $\varphi(x_1, x_2)= \overline{x}_1 \wedge x_2 \vee x_1 \wedge \overline{x}_2$ принадлежит классу $K_л$, так как она представима в виде разложения Жегалкина: $\varphi(x_1, x_2)=0 \oplus 1 \wedge x_1 \oplus 1 \wedge x_2$ ($c_0=0$, $c_1=1$, $c_2=1$). А функция $\varphi(x_1, x_2)=x_1 \wedge x_2$ не допускает такого представления и, следовательно, не принадлежит классу $K_л$.

Класс монотонных функций ($K_м$)

По определению, класс $K_м$ содержит все те функции, для которых справедливо свойство монотонности:

$K_м=\{ f(x_1, x_2, \ldots, x_n) | ((\sigma_1, \sigma_2, \ldots, \sigma_n) \geq (\sigma_1^*, \sigma_2^*, \ldots, \sigma_n^*)) \rightarrow f(\sigma_1, \sigma_2, \ldots, \sigma_n) \geq f(\sigma_1^*, \sigma_2^*, \ldots, \sigma_n^*)\}$. При этом сравнение двух наборов значений переменных $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ и $(\sigma_1^*, \sigma_2^*, \ldots, \sigma_n^*)$ предполагает, что для каждой i-ой переменной $(1 \le i \le n)$ справедливо $\sigma_i \ge \sigma_i^*$. Примером монотонной функции является конъюнкция, а примером немонотонной функции – сложение Жегалкина.

Класс самодвойственных функций ($K_с$)

Функция $f^*(x_1, x_2, \ldots, x_n)$ является двойственной для функции $f(x_1, x_2, \ldots, x_n)$, если $f^*(x_1, x_2, \ldots, x_n) = \overline{f(\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n)}$. Таким образом, двойственная функция получается путем инверсии исходной функции и замены всех переменных на их отрицания.

По определению, класс $K_с$ содержит все те функции, для которых справедливо $f(x_1, x_2, \ldots, x_n) = \overline{f(\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n)}$: $K_с = \{f(x_1, x_2, \ldots, x_n) \mid f(x_1, x_2, \ldots, x_n) = \overline{f(\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n)}\}$.

В качестве примера приведена самодвойственная функция $\psi(x_1, x_2, x_3) = \overline{x}_1 \wedge (\overline{x}_2 \vee \overline{x}_3 \vee x_2 \wedge x_3)$. Двойственная для нее – $\psi^*(x_1, x_2, x_3) = \overline{\overline{\overline{x}}_1 \wedge (\overline{\overline{x}}_2 \vee \overline{\overline{x}}_3 \vee \overline{x}_2 \wedge \overline{x}_3)}$. Эквивалентные преобразования показывают, что $\psi(x_1, x_2, x_3) = \psi^*(x_1, x_2, x_3)$.

5.3. Функционально полные системы логических функций

Рассмотренные в предыдущем разделе классы логических функций позволяют сформулировать критерий функциональной полноты для логических функций, для функций из E_2.

<u>ТЕОРЕМА</u> (критерий Поста–Яблонского).

Система логических функций $S = \{\varphi_1(x_1, x_2, \ldots, x_{n1}), \varphi_2(x_1, x_2, \ldots, x_{n2}), \ldots, \varphi_k(x_1, x_2, \ldots, x_{nk})\}$ функционально полна в E_2, если она:

- содержит хотя бы одну функцию, не принадлежащую классу K_0;
- содержит хотя бы одну функцию, не принадлежащую классу K_1;
- содержит хотя бы одну функцию, не принадлежащую классу $K_л$;
- содержит хотя бы одну функцию, не принадлежащую классу $K_м$;
- содержит хотя бы одну функцию, не принадлежащую классу $K_с$.

Эта теорема позволяет проверять любые системы функций на полноту. Так, легко убедиться, что система функций $S=\{\varphi_1(x_1, x_2)=x_1 \vee x_2, \varphi_2(x_1, x_2)=x_1 \wedge x_2, \varphi_3(x)=\overline{x}\}$, которые порождают алгебру логики, функционально полна. Действительно, функция φ_3 не принадлежит классам K_0, K_1 и $K_м$, а функции $\varphi_1(x_1, x_2)$ и $\varphi_2(x_1, x_2)$ не принадлежат классам $K_л$ и $K_с$. И, таким образом, для данной системы функций справедливы условия теоремы. Она функционально полна.

Надо отметить, что введение дополнительных функций в функционально полную систему не приводит к потере функциональности. Напротив, удаление из такой системы одной или нескольких функций может приводить к потере свойства функциональной полноты. Как отмечалось выше, минимальная по числу функций функционально полная логическая система называется логическим базисом. Вопрос: является ли функционально полная система функций $S=\{\varphi_1(x_1, x_2)=x_1 \vee x_2, \varphi_2(x_1, x_2)=x_1 \wedge x_2, \varphi_3(x)=\overline{x}\}$ логическим базисом? Ответ на этот вопрос – отрицательный. Свойство функциональной полноты системы сохраняется, если из нее удалить $\varphi_1(x_1, x_2)$ либо $\varphi_2(x_1, x_2)$. Ответ очевиден, если учесть свойство де Моргана:

$$x_1 \vee x_2 = \overline{\overline{x_1 \vee x_2}} = \overline{\overline{x}_1 \wedge \overline{x}_2} \qquad \varphi_1(x_1, x_2)=\varphi_3(\varphi_2(\varphi_3(x_1)\varphi_3(x_2)));$$

$$x_1 \wedge x_2 = \overline{\overline{x_1 \wedge x_2}} = \overline{\overline{x}_1 \vee \overline{x}_2} \qquad \varphi_2(x_1, x_2)=\varphi_3(\varphi_1(\varphi_3(x_1)\varphi_3(x_2))).$$

<u>Замечание</u>. Хотя система функций $S=\{\varphi_1(x_1, x_2)=x_1 \vee x_2, \varphi_2(x_1, x_2)= =x_1 \wedge x_2, \varphi_3(x)=\overline{x}\}$ формально не является базисной, отдавая дань уважения одному из основоположников математической логики Д. Булю, этот набор называют *булевым базисом*.

К базисам относятся также следующие наборы:

- базис Шеффера $S_ш=\{\varphi_ш(x_1, x_2)=x_1 | x_2=\overline{x_1 \wedge x_2}\}$, соответствующую операцию «|» называют «штрих Шеффера»;
- базис Вебба $S_в=\{\varphi_в(x_1, x_2)=x_1 \circ x_2=\overline{x_1 \vee x_2}\}$, соответствующую операцию «$\circ$» называют «операция Вебба»;
- импликативный базис $S=\{\varphi_1(x_1, x_2)=x_1 \rightarrow x_2, \varphi_2(x)=\overline{x}\}$.

Существует еще ряд базисов, которые отличаются от перечисленных как набором функций, так и числом входных переменных. Так, функция Шеффера на k входов ($k=3$) образует базис

$\varphi_{\text{ш}}(x_1, x_2, x_3)=\overline{x_1 \wedge x_2 \wedge x_3}$.

Каждый из логических базисов позволяет выразить любую логическую функцию как суперпозицию функций (операций) базиса. Ниже рассмотрены примеры подобных задач.

1. Выразить функции булева базиса в базисе Шеффера с двумя входами.

- Отрицание: $\overline{a}=\varphi_{\text{ш}}(a, a)$.
- Конъюнкция: $\overline{a \wedge b}=\varphi_{\text{ш}}(a, b)$. В предыдущей строке показано, как вычисляется отрицание. Отрицание от левой и правой частей дает

$\overline{\overline{a \wedge b}}= a \wedge b=\overline{\varphi_{\text{ш}}(a,\ b)}=\varphi_{\text{ш}}(\varphi_{\text{ш}}(a,\ b),\ \varphi_{\text{ш}}(a,\ b))$.

- Дизъюнкция: учитывая, что $\varphi_{\text{ш}}(a,\ b)= \overline{a \wedge b}=\overline{a}\vee \overline{b}$, дизъюнкция $a \vee b=\varphi_{\text{ш}}(\overline{a},\ \overline{b})=\varphi_{\text{ш}}(\varphi_{\text{ш}}(a,\ a),\ \varphi_{\text{ш}}(b,\ b))$.

2. Выразить функции булева базиса через функцию Вебба с тремя входами: $\varphi_{\text{в}}(x_1,\ x_2,\ x_3)=\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3$.

- Отрицание $\overline{a}=\varphi_{\text{в}}(a, a, a)$.
- Дизъюнкция:

$a \vee b= \varphi_{\text{в}}(\overline{a},\ \overline{b},\ \overline{b})=\varphi_{\text{в}}(\varphi_{\text{в}}(a, a, a), \varphi_{\text{в}}(b, b, b), \varphi_{\text{в}}\ (b, b, b))$.

- Конъюнкция:

$a \wedge b=\varphi_{\text{в}}(a,b,b)=\varphi_{\text{в}}(\varphi_{\text{в}}(a,b,b), \varphi_{\text{в}}(a,b,b), \varphi_{\text{в}}\ (b, b, b))$;

3. Представить следующую функцию в виде суперпозиции функции Шеффера с тремя входами: $f(x_1, x_2, x_3, x_4)=$

$=\overline{x}_1 \wedge x_3 \wedge x_4 \vee x_2 \wedge \overline{x}_3 \wedge x_4 \vee x_1 \wedge x_2 \wedge \overline{x}_3 =$

$=\varphi_{\text{ш}}(\overline{\overline{x}_1 \wedge x_3 \wedge x_4}, \overline{x_2 \wedge \overline{x}_3 \wedge x_4}, \overline{x_1 \wedge x_2 \wedge \overline{x}_3}\ )=$

$=\varphi_{\text{ш}}(\varphi_{\text{ш}}(\overline{x}_1, x_3, x_4), \varphi_{\text{ш}}(x_2, \overline{x}_3, x_4), \varphi_{\text{ш}}(x_1, x_2, \overline{x}_3))=$

$=\varphi_{\text{ш}}(\varphi_{\text{ш}}(\varphi_{\text{ш}}(x_1, x_1, x_1), x_3, x_4), \varphi_{\text{ш}}(x_2, \varphi_{\text{ш}}(x_3, x_3, x_3), x_4), \varphi_{\text{ш}}(x_1, x_2,$ $\varphi_{\text{ш}}(x_3, x_3, x_3)))$.

5.4. Логические схемы: анализ и синтез

Одно из практических применений функционально полных логических систем – это решение задач анализа и синтеза логических схем. Электронная промышленность производит ряд структурных электронных компонент, которые скомпонованы в микросхемы. Каждая такая компонента на логическом уровне «вычисляет» ту или иную логическую функцию из определенной функционально полной системы. И именно полнота гарантирует, что любая логическая функция может быть реализована в заданной системе подобных элементов.

На практике наибольшее распространение получили следующие элементы, которые «вычисляют» соответствующую функцию:

- дизъюнктор (or);
- конъюнктор (and);
- инвертор (no);
- элемент Шеффера (nand);
- элемент Вебба (nor);
- элемент «сложение по модулю 2» (xor).

Графическое представление перечисленных типов структурных элементов представлено в табл. 5.1.

Каждый структурный логический элемент имеет несколько входов (на левой стороне) и один выход (на правой стороне).

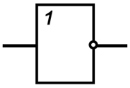
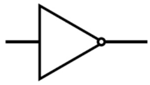
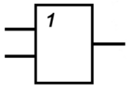
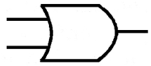
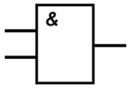
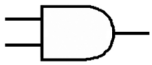
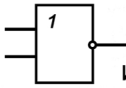
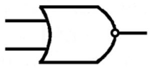
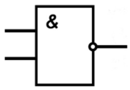
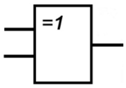
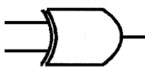
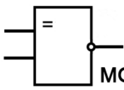
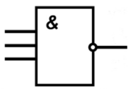
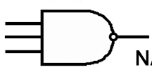
> ***Логическая схема*** – это совокупность структурных логических элементов определенного типа и структура (схема) их соединения.

Схема соединения должна обладать следующими свойствами:

- имеется ровно один элемент, выход которого не соединен ни с одним входом (выход логической схемы);
- имеется ровно n входов, которые не соединены ни с одним выходом, каждому такому входу (вход логической схемы) сопоставляется логическая переменная;

Таблица 5.1

Типы логических структурных элементов

Тип элемента	Российский стандарт (ГОСТ 2.743-91)	Западный стандарт (MIL/ANSI)
Инвертор (NO)	 НЕ	 NO
Дизъюнктор (OR)	 ИЛИ	 OR
Конъюнктор (AND)	 И	 AND
Элемент Вебба на два входа (NOR)	 ИЛИ-НЕ	 NOR
Элемент Шеффера на два входа (NAND)	 И-НЕ	 NAND
Сложение по модулю 2 (XOR)	 MOD2	 XOR
Эквивалентность (NXOR)	 MOD2-НЕ	 NXOR
Элемент Шеффера на три входа (NAND)	 И-НЕ	 NAND

- вход каждого структурного элемента соединен с не более чем одним выходом;
- схема не должна содержать обратные связи.

На рис. 5.1 приведен пример логической схемы, построенной с использованием структурного элемента ИЛИ-НЕ (функция Вебба) на два входа.

С логическими схемами связаны две задачи: задача анализа и задача синтеза логической схемы.

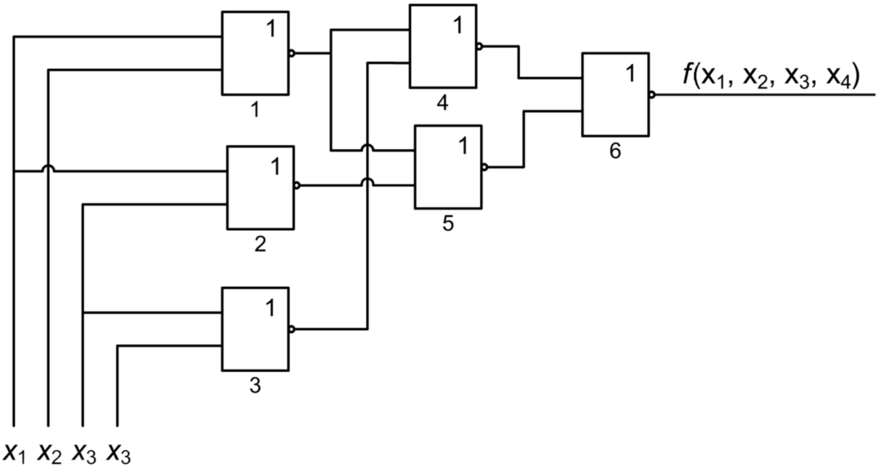


Рис. 5.1. Пример логической схемы

Задача анализа логической схемы

Дано: логическая схема (типы структурных логических элементов и схема их соединения).

Определить: логическую функцию, которая реализуется (вычисляется) схемой.

Решение.

Точка соединения выхода элемента с входом называется узлом схемы. Решение задачи анализа формируется как последовательное восстановление функций, «вычисляемых» в узлах схемы, при перемещении от входов к выходу схемы. Если определены функции на входах структурного элемента, то функция на выходе элемента определяется как его суперпозиция от входных функций. Для схемы (рис. 5.1):

функция на выходе первого элемента: $\overline{x_1 \vee x_2}$;

функция на выходе второго элемента: $\overline{x_1 \vee x_3}$;

функция на выходе третьего элемента: $\overline{x_3 \vee x_4}$;

функция на выходе четвертого элемента: $\overline{\overline{x_1 \vee x_2} \vee \overline{x_3 \vee x_4}}$;

функция на выходе пятого элемента: $\overline{\overline{x_1 \vee x_2} \vee \overline{x_1 \vee x_3}}$;

функция на выходе шестого элемента (выход схемы) –

$\overline{\overline{\overline{x_1 \vee x_2} \vee \overline{x_3 \vee x_4}} \vee \overline{\overline{x_1 \vee x_2} \vee \overline{x_1 \vee x_3}}}$.

После проведения эквивалентных преобразований можно утверждать, что приведенная логическая схема реализует (вычисляет) функцию $f(x_1, x_2, x_3, x_4) = \overline{x}_1 \wedge (\overline{x}_2 \vee \overline{x}_3 \wedge \overline{x}_4)$.

Задача синтеза логической схемы

Дано: логическая функция $f(x_1, x_2, \ldots, x_n)$ и тип (типы) структурных логических элементов синтезируемой схемы.

Определить: число структурных элементов заданного типа и схему их соединения (логическую схему), которая «вычисляет» заданную функцию $f(x_1, x_2, \ldots, x_n)$.

Решение.

Эта задача является обратной по отношению к задаче анализа, и, как следствие, она не имеет однозначного решения. Пространство возможных решений можно сократить за счет введения критерия для задачи. На практике применяют такие критерии, как:

– минимум сложности (числа структурных элементов) логической схемы;

– минимум временных затрат (трудоемкости) поиска решения;

– надежность функционирования синтезируемой логической схемы, что связано с учетом возможных временных задержек в структурных элементах (эффект гонок).

Ниже рассматривается метод синтеза логических схем и приведены примеры его применения для элементов ИЛИ-НЕ (элементы Вебба) и элементов типа И-НЕ (элементы Шеффера) с двумя входами.

Данный метод основан на последовательном разложении исходной функции и представлении ее в виде суперпозиции некоторых входных функций по реализуемой структурным элементом функции. Каждая входная функция также представляется в виде суперпозиции.

Эта процедура применяется к тем входным функциям, которые отличны от переменной.

Замечание. При использовании элемента ИЛИ-НЕ исходную функцию желательно представить в виде минимальной КНФ, а при использовании элемента И-НЕ – в виде минимальной ДНФ. Это обеспечивает, в среднем, минимальность по числу элементов, синтезируемых логических схем.

Пример 1. Синтезировать логическую схему, реализующую функцию $f(x_1,\ x_2,\ x_3)_{\text{КНФ}} = (\overline{x}_1 \vee x_2) \wedge (\overline{x}_1 \vee x_3)$, которая была рассмотрена в разделе 3.3 на элементах типа ИЛИ-НЕ. Эта функция – минимальна в классе КНФ.

Шаг 1. Представление логической функции в виде суперпозиции базисной функции Вебба $\varphi_{\text{в}}(x_1,\ x_2) = x_1 \circ x_2 = \overline{x_1 \vee x_2}$:

$$(\overline{x}_1 \vee x_2) \wedge (\overline{x}_1 \vee x_3) = \overline{\overline{(\overline{x}_1 \vee x_2) \wedge (\overline{x}_1 \vee x_3)}} = \overline{\overline{(\overline{x}_1 \vee x_2)} \vee \overline{(\overline{x}_1 \vee x_3)}},$$

и, следовательно, исходная функция представима в виде суперпозиции $f(x_1,\ x_2,\ x_3) = \varphi_{\text{в}}(\overline{\overline{x}_1 \vee x_2},\ \overline{\overline{x}_1 \vee x_3})$. Полученный результат трактуется следующим образом: для того чтобы на выходе схемы (первый элемент) получилась требуемая функция, необходимо на один вход этого структурного элемента подать функцию $\overline{\overline{x}_1 \vee x_2}$, а на другой – $\overline{\overline{x}_1 \vee x_3}$ (рис. 5.2).

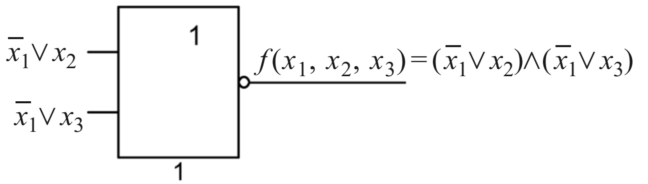


Рис. 5.2. Результат первого шага

Шаг 2. На первый (верхний) вход элемента номер 1 необходимо подать функцию $\overline{\overline{x}_1 \vee x_2}$, которая представима в виде суперпозиции $\varphi_{\text{в}}(\overline{x}_1,\ x_2)$. В результате получается схема, представленная на рис. 5.3.

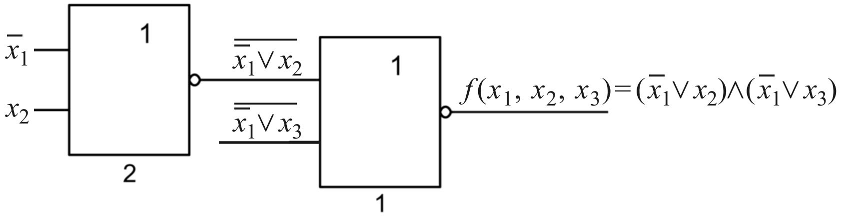


Рис. 5.3. Результат второго шага

Шаг 3. На первый вход второго элемента необходимо подать функцию $\bar{x}_1$, т.е. отрицание от x_1. Как было показано выше, отрицание представляется в виде суперпозиции $\varphi_в(x_1, x_1)$. В результате получается схема, представленная на рис. 5.4.

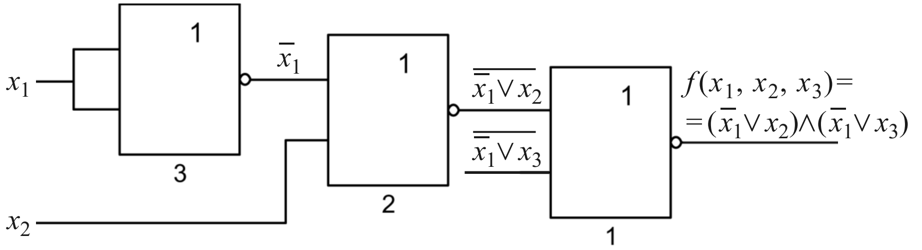


Рис. 5.4. Результат третьего шага

Шаг 4. На второй вход первого элемента необходимо подать функцию $\bar{x}_1 \vee x_3$, которая представима в виде суперпозиции $\varphi_в(\bar{x}_1, x_3)$. В результате получается схема, представленная на рис. 5.5.

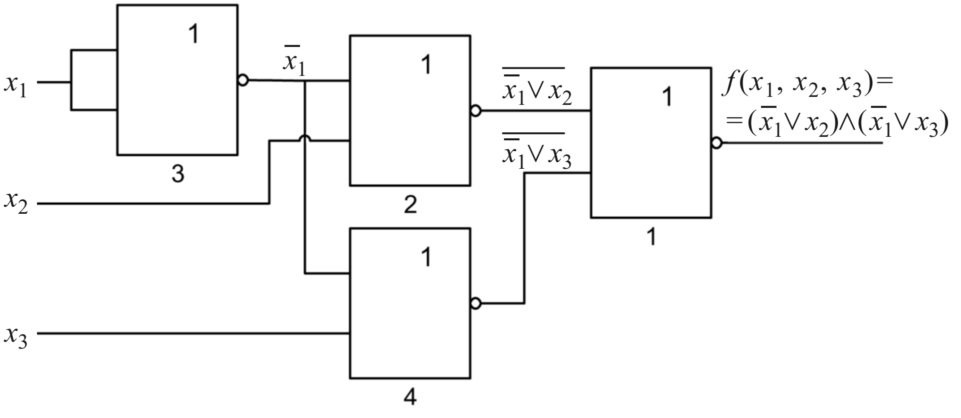


Рис. 5.5. Результат четвертого шага

На верхний вход четвертого элемента необходимо подать функцию $\overline{x}_1$, но эта функция является выходом третьего элемента. А на нижний вход этого элемента подается x_3. Поскольку входам схемы поставлены в соответствие только переменные, эта схема является решением поставленной задачи.

Замечание. Решив для этой схемы задачу анализа, можно убедиться, что она реализует заданную функцию.

Пример 2. Синтезировать логическую схему, реализующую функцию $f(x_1, x_2, x_3)_{ДНФ} = \overline{x}_1 \vee x_2 \wedge x_3$, которая рассмотрена в разделе 3.3 на элементах типа И-НЕ. Эта функция минимальна в классе ДНФ.

Шаг 1. Представление логической функции в виде суперпозиции базисной функции Шеффера $\varphi_ш(x_1, x_2) = x_1 | x_2 = \overline{x_1 \wedge x_2}$:

$$f(x_1, x_2, x_3) = \overline{x}_1 \vee x_2 \wedge x_3 = \overline{\overline{\overline{x}_1 \vee x_2 \wedge x_3}} = \overline{\overline{\overline{x}}_1 \wedge \overline{x_2 \wedge x_3}} = \overline{x_1 \wedge \overline{x_2 \wedge x_3}},$$

и, следовательно, исходная функция представима в виде суперпозиции функции Шеффера $\varphi_ш(x_1, \overline{x_2 \wedge x_3})$. Полученный результат трактуется следующим образом: для того чтобы на выходе схемы (первый элемент) получилась требуемая функция, необходимо на один вход этого структурного элемента подать x_1, а на второй – функцию $\overline{x_2 \wedge x_3}$ (рис. 5.6).

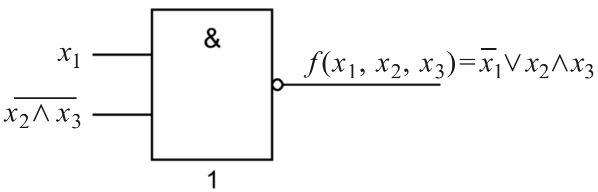


Рис. 5.6. Результат первого шага

Шаг 2. На нижний вход элемента номер 1 необходимо подать функцию $\overline{x_2 \wedge x_3}$, которая представима в виде суперпозиции $\varphi_ш(x_2, x_3)$. В результате получается схема, представленная на рис. 5.7.

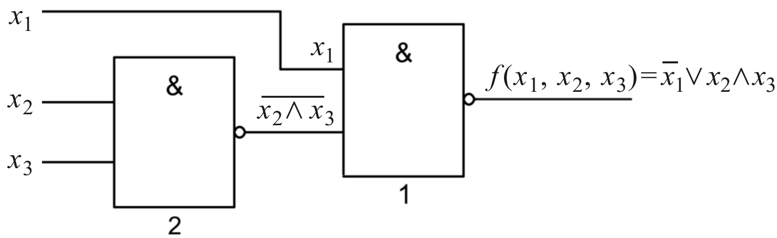


Рис. 5.7. Результат второго шага

Поскольку входам схемы поставлены в соответствие только переменные, эта схема является решением поставленной задачи.

Замечание. Как видно из примеров, решение задачи синтеза логической функции основано на разложении (декомпозиции) функции по заданному логическому базису (переход от выхода элемента к его входам). Этот переход не всегда однозначен.

5.5. Задача анализа и синтеза логических функций на элементах высокой степени интеграции

Современный уровень развития технологии электронных схем позволяет создавать компоненты, в которых на небольшой площади кристалла размещаются тысячи, десятки тысяч логических элементов, соединенных между собой определенным образом. Такие схемы конструктивно выпускаются в виде *интегральных микросхем* различного функционального назначения. Иногда подобные микросхемы называют чипами.

Память – название одного из типов таких микросхем. Память, как и следует из ее названия, предназначена для запоминания, хранения различных данных, цифровых кодов. Память состоит из однотипных элементов – ячеек памяти. Каждое данное (код) хранится в отдельной ячейке памяти. Ключевая функция памяти – выдача этих кодов на выходы микросхемы по запросу извне. Каждая ячейка памяти идентифицируется уникальным адресом, кодом, который подается на вход. Упрощенная структурная модель элемента памяти с прямым доступом (RAM) приведена на рис. 5.8.

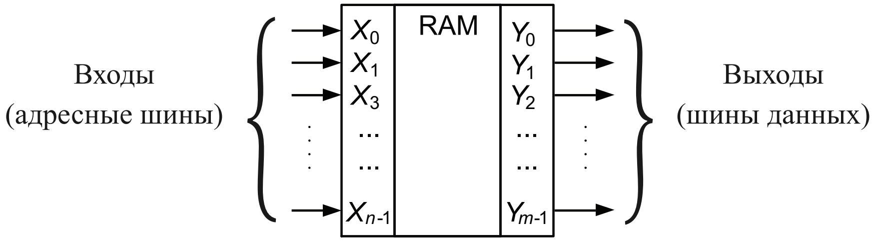


Рис. 5.8. Упрощенная модель элемента памяти

Элемент памяти имеет n входных шин и m выходных шин ((n-m)-память). Входные и выходные шины могут находиться в одном из двух состояний – 0 либо 1. Входные шины задают адрес ячейки памяти, а выходные шины служат для передачи данных, которые находятся в этой ячейке. Если число входов n, то емкость памяти такого элемента равна 2^n, что определяется множеством состояний входных шин. Выходы памяти определяют двоичный вектор длины m.

Рассмотренный элемент памяти позволяет реализовать систему из m различных логических функций от n переменных:

$$\begin{cases} Y_0=f_0(x_0,\ x_1,\ x_2,\ \ldots,\ x_{n\text{-}1}) \\ Y_1=f_1(x_0,\ x_1,\ x_2,\ \ldots,\ x_{n\text{-}1}) \\ \cdot\quad\cdot\quad\cdot\quad\cdot \\ Y_{\text{m-1}}=f_{\text{m-1}}(x_0,\ x_1,\ x_2,\ \ldots,\ x_{n\text{-}1}) \end{cases}$$

Или в векторном виде: $\vec{Y}=\vec{f}(\vec{x})$. В роли логических переменных выступают адресные шины, а в роли логических функций – шины данных.

Задача анализа памяти

Дано: элемент памяти (значения n и m, содержимое ячеек памяти в табличном виде).

Определить: логические функции, которые реализуются (вычисляются) для каждой выходной шины.

Решение.

Для восстановления логической функции $Y_j = f_j(x_0, x_1, x_2, \ldots, x_{n\text{-}1})$, реализуемой на j-ой выходной шине, необходимо:

- в j-ом столбце отметить все строки, в которых $Y_j=1$;
- для каждой отмеченной строки выписать конституенту, соответствующую адресной части этой строки;
- записать СовДНФ искомой функции, взяв дизъюнкцию всех конституент, найденных на предыдущем шаге.

Эта процедура выполняется для каждой функции $Y_j (1 \le j \le m)$.

Пример. Решить задачу анализа для (3-2) элемента памяти, содержимое которого задано в табл. 5.2.

Таблица 5.2

x_1	x_2	x_3	Y_0	Y_1
0	0	0	1	0
0	0	1	0	1
0	1	0	0	0
0	1	1	1	0
1	0	0	0	0
1	0	1	0	1
1	1	0	1	1
1	1	1	0	1

адрес ячейки — данные в ячейке

Функция y_1 принимает значение 1 в трех строках на следующих наборах значений адресных шин: 000, 011, 110. И следовательно, $Y_0 = f_0(x_1, x_2, x_3)_{\text{СовДНФ}} = \overline{x}_1 \wedge \overline{x}_2 \wedge \overline{x}_3 \vee \overline{x}_1 \wedge x_2 \wedge x_3 \vee x_1 \wedge x_2 \wedge x_3$.

Функция y_2 принимает значение 1 в четырех строках на следующих наборах значений адресных шин: 001, 101, 110, 111. И следовательно, $Y_1 = f_1(x_2, x_2, x_3)_{\text{СовДНФ}} = \overline{x}_1 \wedge \overline{x}_2 \wedge x_3 \vee x_1 \wedge \overline{x}_2 \wedge x_3 \vee x_1 \wedge x_2 \wedge \overline{x}_3 \vee x_1 \wedge x_2 \wedge x_3$.

<u>Задача синтеза логических функций для элемента памяти</u>

Дано:

- тип элемента памяти (значения n и m);
- m логических функций $Y_j=f_j(x_1, x_2, \ldots, x_n)$ от n переменных, заданных в произвольном виде.

Определить: содержимое элементов памяти, обеспечивающих вычисление заданных функций.

<u>Замечание</u>. Для записи данных в память существуют специальные устройства, называемые программаторами. Исходные данные для программаторов задаются в виде таблицы, подобной табл. 5.2, которую называют *программой прошивки* памяти. Программатор на основании этой таблицы формирует содержимое ячеек памяти, «*прошивает память*».

<u>Решение</u>.

В отличие от задачи синтеза логической схемы задача синтеза для элементов памяти однозначна (с точностью до перестановки функций по выходным шинам).

Для решения этой задачи необходимо каждую исходную функцию представить в табличном виде, а затем свести их в одну таблицу, совместив их аргументные части. Эта таблица и есть решение задачи синтеза, так как она является программой прошивки заданного элемента памяти.

<u>Пример</u>. Решить задачу синтеза для (3-3) элемента памяти, который реализует следующие три функции:

$$Y_0=f_0(x_1, x_2, x_3)=\overline{\overline{\bar{x}_1\to\bar{x}_2\wedge x_3}\vee x_2\oplus x_3};$$

$$Y_1=f_1(x_1, x_2, x_3)=\overline{\bar{x}_1\wedge\overline{\bar{x}_2\wedge\bar{x}_3}\vee\bar{x}_1\leftrightarrow x_2};$$

$$Y_2=f_2(x_1, x_2, x_3)=x_3\wedge(x_1\to x_2)\wedge(x_2\to x_1).$$

Табличное представление функций приведено в табл. 5.3 – 5.5.

Таблица 5.3

x_1	x_2	x_3	Y_0
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Таблица 5.4

x_1	x_2	x_3	Y_1
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Таблица 5.5

x_1	x_2	x_3	Y_2
0	0	0	0
0	0	1	0
0	1	1	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Решение задачи в виде таблицы прошивки памяти приведено в табл. 5.6.

Таблица 5.6

Программа прошивки

x_1	x_2	x_3	Y_0	Y_1	Y_2
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	1
1	0	0	1	0	0
1	0	1	0	0	1
1	1	0	0	1	0
1	1	1	1	1	0
адрес ячейки			данные в ячейке		

- **Задачи и упражнения**

① Представить приведенные абстрактные функции как суперпозицию системы функций $S=\{\varphi(x)=\uparrow x, \psi(x_1, x_2)=x_1\downarrow x_2\}$:

- $\lambda_1(x_1, x_2)=(\uparrow x_1)\downarrow(\uparrow x_1)$;
- $\lambda_2(x_1, x_2)=(\uparrow x_1)\downarrow(\uparrow x_2)$;
- $\lambda_3(x_1, x_2)=\uparrow(x_1\downarrow x_2)$;
- $\lambda_4(x_1, x_2)=(\uparrow x_1)\downarrow(\uparrow(\uparrow(x_1\downarrow x_2)))$;
- $\lambda_5(x_1, x_2, x_3)=\uparrow((\uparrow(x_1\downarrow x_2))\downarrow(\uparrow x_3))$.

② Доказать, что функция $g_1(x_1, x_2, x_3)=(x_1\leftrightarrow x_3)\leftrightarrow x_2$ является линейной функцией.

③ Доказать, что функция $g_2(x_1, x_2, x_3)=x_2\wedge(x_1\leftrightarrow x_3)$ является монотонной функцией.

④ Доказать, что функция $g_3(x_1, x_2, x_3)=\vee(0, 1, 4)$ удовлетворяет критерию функциональной полноты в E2.

⑤ Представить функцию $g_4(x_1, x_2, x_3)=\overline{x_2\wedge(x_1\vee\overline{x_3})}$ как суперпозицию функции Шеффера наиболее компактным способом. Построить минимальную логическую схему на элементах И-НЕ, реализующую заданную функцию.

⑥ Представить функцию Шеффера как суперпозицию функции Вебба (Пирса).

⑦ Построить минимальную логическую схему на элементах И-НЕ, реализующую функцию $\overline{x_2}\vee((x_1|x_3)|x_2)$.

6. МАТЕМАТИЧЕСКАЯ ЛОГИКА: ЛОГИКА ПРЕДИКАТОВ

6.1. Понятие предиката и его свойства

Пусть $I=M_1\times M_2\times\ldots\times M_n=\{(m_1,m_2, \ldots m_n)|m_i\in M_i, i\in\{1, 2, \ldots n\}\}$ $(I=M^n)$ – некоторое n-арное декартово произведение.

Предикат $P(x_1, x_2, \ldots, x_n)$, определенный на области I, есть функция, отображающая $M_1\times M_2\times\ldots\times M_n\ (M^n)$ в $\{0, 1\}$, $P(x_1, x_2, \ldots, x_n)$: $M_1\times M_2\times\ldots\times M_n\rightarrow\{0, 1\}$.

Предикат может принимать значение 1 (истина) либо 0 (ложь). Множество I называют *областью определения* предиката $P(x_1, x_2, \ldots, x_n)$, а его подмножество, для каждого элемента которого предикат принимает значение 1, называют *областью истинности предиката P* и обычно обозначают как I_P. Для элементов множества $I\setminus I_P$ предикат принимает значение 0.

• Если $I_P=I$, предикат называют тождественно истинным: $P(x_1, x_2, \ldots, x_n)=1$ на всей области определения.

• Если $I_P=\varnothing$, предикат называют тождественно ложным: $P(x_1, x_2, \ldots, x_n)=0$ на всей области определения.

• Если $I_P\neq I$ и $I_P\neq\varnothing$, предикат называют выполнимым.

Число переменных определяет «местность» предиката (n-местный предикат). Если $n=0$, предикат является логическим высказыванием, значение которого истина либо ложь.

Иногда предикаты называют логическими высказываниями с переменными. Действительно, ранее рассмотренное высказывание «Город стоит на берегу реки» нельзя отнести к логическому, так как нельзя определить меру его истинности. Но высказывание $P(x, y)$=«город x стоит на берегу реки y», где x принадлежит множеству городов, а y принадлежит множеству рек, является двуместным предикатом. Значение истинности этого предиката зависит от конкретных значений переменных x и y. Так, при x=«Москва» и y=«Москва» значение P(Москва, Москва) равно 1. А значение предиката P(Париж, Темза)=0.

К предикатам, как к объектам математической логики, применимы все известные логические операции: дизъюнкция, конъюнкция, отрицание, следования, эквивалентности и сложение Жегалкина. Выполнение этих операций приводит к изменению области истинности предикатов.

Пусть на некотором универсуме I определены два n-местных предиката $P(x_1, x_2, \dots, x_n)$ и $Q(x_1, x_2, \dots, x_n)$, область истинности которых I_P и I_Q соответственно.

Дизъюнкция предикатов $P(x_1, x_2, \dots, x_n)$ и $Q(x_1, x_2, \dots, x_n)$ есть предикат $P(x_1, x_2, \dots, x_n) \vee Q(x_1, x_2, \dots, x_n)$, область истинности которого $I_{P\vee Q}=I_P \cup I_Q$.

Конъюнкция предикатов $P(x_1, x_2, \dots, x_n)$ и $Q(x_1, x_2, \dots, x_n)$ есть предикат $P(x_1, x_2, \dots, x_n) \wedge Q(x_1, x_2, \dots, x_n)$, область истинности которого $I_{P\wedge Q}=I_P \cap I_Q$.

Отрицание предиката $P(x_1, x_2, \dots, x_n)$ есть предикат $\overline{P(x_1, x_2, \dots, x_n)}$, область истинности которого $I_{\bar{P}} = I \setminus I_P$.

Логическое следование из $P(x_1, x_2, \dots, x_n)$ предиката $Q(x_1, x_2, \dots, x_n)$ есть предикат $P(x_1, x_2, \dots, x_n) \rightarrow Q(x_1, x_2, \dots, x_n)$, область истинности которого $I_{P\rightarrow Q} = \overline{I_P} \cup I_Q$.

Эквивалентность предикатов $P(x_1, x_2, \dots, x_n)$ и $Q(x_1, x_2, \dots, x_n)$ есть предикат $P(x_1, x_2, \dots, x_n) \leftrightarrow Q(x_1, x_2, \dots, x_n)$, область истинности которого $I_{P\leftrightarrow Q}=\overline{I_P \setminus I_Q \cup I_Q \setminus I_P}$.

Сложение по модулю 2 предикатов $P(x_1, x_2, \ldots, x_n)$ и $Q(x_1, x_2, \ldots, x_n)$ есть предикат $P(x_1, x_2, \ldots, x_n) \oplus Q(x_1, x_2, \ldots, x_n)$, область истинности которого $I_{P\oplus Q} = I_P \backslash I_Q \cup I_Q \backslash I_P$.

Если в предикат вместо переменных подставить их конкретные значения из области определения, предикат превращается в логическое высказывание, которое либо истинно, либо ложно.

Для предикатов, кроме известных логических операций, определены две кванторные операции – квантор существования ($\exists$) и квантор всеобщности ($\forall$).

По определению, предикат $\exists x_i, P(x_1, x_2, \ldots, x_i, \ldots, x_n)$ есть дизъюнкция предикатов $P(x_1, x_2, \ldots, x_{i-1}, a_1, x_{i+1}, \ldots, x_n) \vee$
$\vee P(x_1, x_2, \ldots, x_{i-1}, a_2, x_{i+1}, \ldots, x_n) \vee \ldots \vee P(x_1, x_2, \ldots, x_{i-1}, a_k, x_{i+1}, \ldots, x_n) \vee \ldots$, в каждый из которых вместо переменной x_i подставлено допустимое ее значение (константа) из области определения. Результат выполнения этой кванторной операции не зависит от переменной x_i. Предикат $\exists x_i, P(x_1, x_2, \ldots, x_i, \ldots, x_n)$ принимает значение «истина», когда хотя бы одно из логических слагаемых равно 1.

По определению, предикат $\forall x_i, P(x_1, x_2, \ldots, x_i, \ldots, x_n)$ есть конъюнкция предикатов $P(x_1, x_2, \ldots, x_{i-1}, a_1, x_{i+1}, \ldots, x_n) \wedge$
$\wedge P(x_1, x_2, \ldots, x_{i-1}, a_2, x_{i+1}, \ldots, x_n) \wedge \ldots \wedge P(x_1, x_2, \ldots, x_{i-1}, a_k, x_{i+1}, \ldots, x_n) \wedge \ldots$, в которых вместо переменной x_i подставлены допустимые ее значения (константы) из области определения. Результат выполнения этой кванторной операции не зависит от переменной x_i. Предикат $\forall x_i, P(x_1, x_2, \ldots, x_i, \ldots, x_n)$ принимает значение «истина», когда все логические сомножители принимают значение 1.

Для простого случая, когда область определения одноместного предиката $P(x)$ задана конечным множеством M, например $M=\{a, b, c\}$, предикаты $\forall x, P(x)$ и $\exists x, P(x)$ представимы как

$\forall x, P(x) = P(a) \wedge P(b) \wedge P(c)$;

$\exists x, P(x) = P(a) \vee P(b) \vee P(c)$, где $P(a)$, $P(b)$ и $P(c)$ – логические высказывания, значение которых равно 1 либо 0.

Предикат, на который распространяется действие квантора по некоторой переменной, называют *областью действия квантора* по этой переменной. Если некоторая переменная входит в область действия квантора по этой переменной, то она *связана* этим квантором. Если в предикате все переменные связаны, то говорят, что предикат *замкнут*. Замкнутый предикат является логическим высказыванием без переменных и принимает значение 1 либо 0.

Переменную, которая не входит в область действия квантора по этой переменной, называют *свободной* переменной. Не допускается, чтобы одна и та же переменная в предикате была и свободной, и связанной.

Предикат $\forall x,\ P(x) \vee Q(x, y)$ является недопустимым, так как здесь переменная x одновременно связана и свободна.

Два предиката $F(x_1, x_2, \dots, x_n)$ и $G(x_1, x_2, \dots, x_n)$ *равносильны на некотором множестве I*, если при любой интерпретации этих формул они принимают одинаковые логические значения при всех значениях переменных $x_1, x_2, \dots, x_n$ из области I.

Кванторные операции могут применяться к многоместным предикатам по различным переменным. Так, двуместный предикат $P(x, y)$ допускает следующие комбинации кванторных операций: $\forall x \forall y P(x, y)$, $\forall x \exists y P(x, y)$, $\exists x \forall y P(x, y)$, $\exists x \exists y P(x, y)$, $\forall x \forall y P(x, y)$, $\forall y \exists x P(x, y)$, $\exists y \forall x P(x, y)$, $\exists y \exists x P(x, y)$.

Пример 1. Пусть предикат $P(x, y)$= «$x \geq y$» определен на множестве N^2, где N – множество натуральных чисел. Тогда предикат $\forall x \exists y P(x, y)$ (читается как «для любого x найдется такое y, что $x \geq y$») равен 1.

А предикат $\forall x \forall y P(x, y)$ для этого примера равен 0.

Пример 2. Записать предикат $Q(i, j)$, принимающий значение «истина» для всех элементов квадратной матрицы $n\times n$, которая определяется ее затененной частью (включая побочную диагональ) на рис. 6.1.

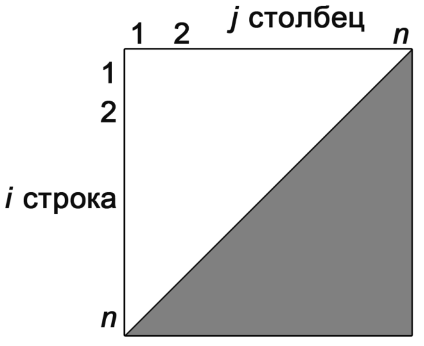


Рис. 6.1. Матрица $n\times n$

Побочная диагональ матрицы определяется уравнением $i=n-j+1$, $1\le i\le n$ и $1\le j\le n$.

Затененная часть матрицы расположена ниже побочной диагонали, и следовательно, для этой части матрицы справедливо неравенство $n-j+1\le i$. И таким образом, предикат $Q(i, j)=(n-j+1\le i)\wedge(1\le i)\wedge$ $\wedge(i\le n)\wedge(1\le j)\wedge(j\le n)$.

Любая функция также может быть представлена в виде предиката. Так, функцию $y=x^2$ можно задать предикатом $P(x, y)=$«$y=x^2$». Для него $P(3, 9)=1$, а $P(4,7)=0$. Использование функций в предикатах требует более точного определения предиката путем введения понятия формулы логики предикатов.

6.2. Формула логики предикатов

Для определения формулы логики предикатов необходимо определить символьный базис (алфавит), а также правила образования из него формул. Такая технология определения объектов широко применяется в формальных теориях (исчислениях).

Для формул логики предикатов определены следующие наборы символов:

– Множество *предметных символов* $x, y, z, x_1, x_2, \ldots, y_1, y_2$ и т.д., строчные символы из последней части латинского алфавита с индексами или без индексов.

– Множество *предметных констант* $a, b, c, a_1, a_2, .., b_1, b_2$ и т.д., строчные символы из начальной части латинского алфавита с индексами или без индексов. Предметные константы определяют множество возможных значений предметных переменных.

– Множество *функциональных символов* $f, g, h, f_1, f_2, \ldots, g_1, g_2$ и т.д., строчные символы из средней части латинского алфавита с индексами или без.

– *Предикатные символы* $P, Q, F, P_1, P_2, \ldots, G_1, G_2$ и т.д., прописные символы из средней части латинского алфавита с индексами или без.

– Множество *логических связок*, операций $\wedge, \vee, \bar{\ }, \rightarrow, \leftrightarrow, \oplus, \exists, \forall$.

– Множество *вспомогательных символов* – скобки, запятая.

Из этих наборов символов создаются последовательности символов. Построенные по определенным правилам последовательности объявляются формулами логики предикатов. Для определения понятия формулы логики предикатов вводятся два вспомогательных понятия – терм и элементарная формула.

Определение ***терма*** (рекурсивное):

– всякий предметный символ или предметная константа есть *терм*;

– если f – функциональный символ, а $\lambda_1, \lambda_2, \ldots, \lambda_k$ – термы, то $f(\lambda_1, \lambda_2, \ldots, \lambda_k)$ есть терм;

– выражение является термом, если оно получено в результате конечного числа применения предыдущих двух правил.

Определение ***элементарной формулы логики*** предикатов: «Если $\lambda_1, \lambda_2, \ldots, \lambda_k$ – термы, а P – предикатный символ, то выражение $P(\lambda_1, \lambda_2, \ldots, \lambda_k)$ есть элементарная формула».

Определение ***формулы логики*** предикатов (рекурсивное):

– всякая элементарная формула есть формула логики предикатов;

– если A и B, $A(x)$ и $B(x)$ – формулы логики предикатов, тогда $(A \wedge B)$, $(A \vee B)$, $(\overline{A})$, $(A \rightarrow B)$, $(A \oplus B)$, $(A \leftrightarrow B)$, $(\forall x)(A(x))$, $(\exists x)(A(x))$ есть формулы логики предикатов;

– выражение является формулой логики предикатов, если оно получено в результате конечного числа применения предыдущих двух правил.

Замечание. в определении формулы используется много скобок. Это делается для однозначного определения структуры формулы. Иногда скобки опускают, если это не приводит к искажению структуры формулы.

Для формул, так же как и для предикатов, определены понятия области действия квантора, понятие свободной и понятие связанной переменных.

Формула логики предикатов не является эквивалентом понятия предиката. Для того чтобы формула превратилась в предикат, необходимо провести ее *интерпретацию* $\Im$.

Под ***интерпретацией*** формулы логики предикатов с областью интерпретации M понимается совокупность функций, которая удовлетворяет следующим условиям:

– каждой предметной константе a_i однозначно сопоставлен элемент $\Im(a_i)$ из M;

– каждому n-местному терму f сопоставлена функция $\Im(\mathrm{f})$: $M^n \rightarrow M$;

– каждому n-местному предикатному символу P сопоставлено n-арное отношение $\Im(P)$, такое, что $\Im(P) \subseteq M^n$.

Пример. Дана формула $\forall x, P(f(x), y)$. Ниже приведена одна из возможных ее интерпретаций:

– область возможных (I) значений переменных x и y – множество действительных чисел;

– функциональный символ (терм $f(x)$) интерпретируется как квадратичная функция $f(x)=x^2-15x+20$;

– предикатный символ интерпретируется как $P(x,y)=$ «$x \leq y$».

При такой интерпретации формула $\forall x, P(f(x), y)$ превращается в предикат $\forall x, P(x^2-15x+20, y)=\forall x, (x^2-15x+20\leq y)$.

Две формулы F_1 и F_2 логики предикатов ***равносильны в области определения M***, если при любой интерпретации они принимают одинаковые значения истинности на всех наборах значений переменных из области M.

Две формулы F_1 и F_2 логики предикатов ***равносильны***, если они равносильны на любой области значений переменных.

Для формул логики предикатов справедливы все равносильности, известные для логики высказываний, такие как идемпотентность, коммутативность, ассоциативность, дистрибутивность, де Моргана, двойное отрицание и все производные от них равносильности.

Дополнительные равносильности связаны с введением кванторов всеобщности и существования:

- $\overline{\forall x F(x)}=\exists x \overline{F(x)}$;
- $\overline{\exists x F(x)}=\forall x \overline{F(x)}$;
- $\forall x(P(x, y)\wedge Q(x, z))=\forall x P(x, y)\wedge\forall x Q(x, z)$;
- $\exists x(P(x, y)\vee Q(x, z))=\exists x P(x, y)\vee\exists x Q(x, z)$;
- $\forall x\forall y P(x, y)=\forall y\forall x P(x, y)$;
- $\exists x\exists y P(x, y)=\exists y\exists x P(x, y)$;
- $\forall x(C\wedge P(x, y))=C\wedge\forall x P(x, y)$;
- $\exists x(C\wedge P(x, y))=C\wedge\exists x P(x, y)$;
- $\forall x(C\vee P(x, y))=C\vee\forall x P(x, y)$;

- $\exists x(C \vee P(x, y)) = C \vee \exists x P(x, y)$;
- $\forall x(C \to P(x, y)) = C \to \forall x P(x, y)$;
- $\exists x(C \to P(x, y)) = C \to \exists x P(x, y)$.

В приведенных равенствах C – формула, не содержащая переменную x в качестве свободной. Такие формулы иногда называют константными относительно переменной x.

Пример.

Упростить формулу $\forall x P(x) \vee \exists x \overline{P(x)} \wedge \forall y Q(y)$.

Решение.

$$\forall x P(x) \vee \exists x \overline{P(x)} \wedge \forall y Q(y) = \forall x P(x) \vee \overline{\forall x P(x)} \wedge \forall y Q(y) =$$
$$= (\forall x P(x) \vee \overline{\forall x P(x)}) \wedge (\forall x P(x) \vee \forall y Q(y)) = 1 \wedge (\forall x P(x) \vee \forall y Q(y)) =$$
$$= \forall x P(x) \vee \forall y Q(y).$$

Для формул логики предикатов определено правило переименования переменных. Одна связанная переменная в формуле может быть заменена другой связанной переменной одновременно во всей области действия квантора по этой переменной и в самом кванторе. Полученная таким образом формула равносильна исходной формуле. В приведенном примере связанная переменная x в формуле $\exists x,\ Q(x)$ заменена на связанную переменную y:

$$\exists x,\ P(x) \wedge \exists x,\ Q(x) = \exists x,\ P(x) \wedge \exists y,\ Q(y).$$

6.3. Общезначимость и выполнимость формул

Формула F логики предикатов ***выполнима в области M***, если существует такая интерпретация этой формулы и такие значения предметных переменных из области M, при которых формула F принимает значение 1.

Формула F логики предикатов ***выполнима***, если существует область M и такая интерпретация, при которых формула F выполнима.

Формула F логики предикатов является ***тождественно истинной в области M***, если при любой интерпретации и при любых значениях предметных переменных из области M формула F принимает значение 1.

Формула F логики предикатов ***общезначима***, если она тождественно истинна на любой области M.

Формула F логики предикатов является ***тождественно ложной*** в области M, если она принимает значение 0 при любой интерпретации и при любых значениях предметных переменных из области M.

Задачи проверки формул на выполнимость и на общезначимость, в отличие от формул логики высказываний, относятся к классу трудно решаемых задач. Если область определения конечна, эти задачи разрешимы. В этом случае проверка выполнимости и общезначимости сводится к построению и анализу таблиц истинности.

Примеры.

1. Доказать выполнимость формулы $F = \forall x \exists y, Q(x, y)$.

Для решения этой задачи необходимо найти подходящую интерпретацию, при которой $F = 1$.

Пусть K – множество четных чисел, а область определения I формулы $I = K \times K$.

Пусть $Q(x, y) =$ «x ***mod*** $y = 0$» (x делится нацело, без остатка, на y).

При такой интерпретации для каждого четного x из области определения существует его делитель y. В частности, таким делителем для любого x является сам x. При такой интерпретации формула $F = 1$, и следовательно, она выполнима.

2. Доказать общезначимость формулы

$A = \forall x, Q(x, y) \rightarrow \exists x, Q(x, y)$.

При доказательстве используются равносильности логики высказываний и логики предикатов:

$$\forall x,\ Q(x, y)\to\exists x,\ Q(x, y)=\overline{\forall x,\ Q(x, y)}\vee\exists x,\ Q(x, y)=\exists x,\ \overline{Q(x, y)}\vee$$
$$\vee\exists x,\ Q(x, y)=\exists x,\ (\overline{Q(x, y)}\vee Q(x, y))=\exists x,\ (1)=1.$$

3. Доказать общезначимость формулы:

$$B=\exists x,\ P(x)\wedge\overline{\exists x,\ \overline{Q(x)}}\to\forall x,\ Q(x)\wedge\overline{\exists x,\ \overline{R(x)}}\vee\exists x,\ P(x)\wedge\exists x,\ \overline{R(x)}.$$

Действительно, после преобразования импликации и использования равносильностей для отрицания предикатов, свойств дистрибутивности и констант 0 и 1 исходная формула сводится к тождеству:

$$\exists x,\ P(x)\wedge\overline{\exists x,\ \overline{Q(x)}}\to\forall x,\ Q(x)\wedge\overline{\exists x,\ \overline{R(x)}}\vee\exists x,\ P(x)\wedge\exists x,\ \overline{R(x)}=$$
$$=\overline{\exists x,\ P(x)\wedge\overline{\exists x,\ \overline{Q(x)}}}\vee\forall x,\ Q(x)\wedge\overline{\exists x,\ \overline{R(x)}}\vee\exists x,\ P(x)\wedge\exists x,\ \overline{R(x)}=$$
$$=\overline{\exists x,\ P(x)}\vee\overline{\overline{\exists x,\ \overline{Q(x)}}}\vee\forall x,\ Q(x)\wedge\overline{\exists x,\ \overline{R(x)}}\vee\exists x,\ P(x)\wedge\exists x,\ \overline{R(x)}=$$
$$=\overline{\exists x,\ P(x)}\vee\overline{\forall x,\ Q(x)}\vee\forall x,\ Q(x)\wedge\overline{\exists x,\ \overline{R(x)}}\vee\exists x,\ \overline{R(x)}=$$
$$=\overline{\exists x,\ P(x)}\vee\overline{\forall x,\ Q(x)}\vee(\overline{\exists x,\ \overline{R(x)}}\vee\exists x,\ \overline{R(x)})=\overline{\exists x,\ P(x)}\vee\overline{\forall x,\ Q(x)}\vee 1=1.$$

• Задачи и упражнения

① Изобразить на координатной плоскости области истинности следующих предикатов:

а) $\overline{x>2}\wedge(x<y)$;

б) $(|x|\leq 4)\wedge(|y|\leq 4)\wedge(x\times y<0)$;

в) $(|x|\leq 4)\wedge\overline{x<0}$;

г) $((x>2)\vee(y>1))\wedge((x<-1)\vee(y<-2))$.

② На множестве M задано некоторое бинарное отношение $R \subseteq M \times M$. На M определен одноместный предикат $P(x)=$«$x \in M$». На R определен двуместный предикат $Q(x, y)=$«$(x, y) \in R$». Используя кванторные операции и другие логические операции, записать предикат, который принимает значение «истина», если:

а) отношение R – не рефлексивно;

б) отношение R – иррефлексивно;

в) отношение R – антисимметрично;

г) отношение R – не антисимметрично;

д) отношение R – не транзитивно;

е) отношение R – интранзитивно.

③ На декартовом произведении $M \times M$, где $M=\{a, b, c\}$, определены два двуместных предиката $P(x, y)$ и $Q(x, y)$. Представить следующие формулы без использования кванторных операций.

а) $\exists x,\ Q(x) \wedge \forall z,\ P(y, z)$;

б) $\forall x \exists y,\ (P(x, y) \vee Q(y, z))$;

в) $\forall x \exists y,\ P(x, y) \rightarrow \exists y \forall x,\ Q(x, y)$.

④ Проверить на общезначимость следующие формулы логики предикатов:

а) $\forall x,\ P(x) \rightarrow \exists x,\ Q(x)$;

б) $\forall x,\ Q(x) \vee \forall x,\ P(x) \rightarrow \forall x,\ (Q(x) \vee P(x))$;

в) $\forall x \forall y,\ Q(x, y) \rightarrow \exists y \exists x,\ Q(x, y)$.

7. КОНЕЧНЫЕ АВТОМАТЫ

7.1. Определение конечного автомата

Под *автоматом* понимаются специального вида математические функции, которые адекватно описывают процессы, протекающие в различного рода автоматических системах. В приложениях именно такие устройства принято называть автоматами. Однако «автомат» как устройство не идентичен «автомату» как математическому объекту. Скорее можно говорить, что математическое понятие «автомат» является подходящей математической моделью для описания широкого класса автоматических устройств.

Пусть

- $X=\{x_1, x_2, x_3, \ldots, x_n\}$ – конечное множество абстрактных символов, называемое *входным алфавитом*;
- $Y=\{y_1, y_2, y_3, \ldots, y_m\}$ – конечное множество абстрактных символов, называемое *выходным алфавитом*;
- $Q=\{q_1, q_2, q_3, \ldots, x_k\}$ – конечное множество абстрактных символов, называемое *алфавитом внутренних состояний*.

Словом в алфавите X называют последовательность, составленную из символов этого алфавита. Подобным образом определяются слова в алфавите Y. Число букв в слове ξ называют его длиной и обозначают как $l(\xi)$. Для указания слова над алфавитом X, длина которого равна $s(s\geq 1)$, будет использоваться обозначение $\xi_x^s=x(1)x(2)x(3)\ldots x(s)$. В этой строке $x(i)$ есть символ из алфавита X, стоящий в i-й позиции слова ξ_x^s $(1\leq i\leq s)$. Слово, длина которого равна нулю, есть пустое слово (обозначается символом λ). Пустое слово не содержит ни одного символа. Оно не зависит от алфавита.

Пусть X^*, Y^* – множество всех возможных слов над алфавитами X и Y соответственно. В теории автоматов рассматривают функции φ: $X^* \rightarrow Y^*$, которые допускают конечное описание.

Совокупность $A=\langle X, Y, Q, G, F\rangle$, в которой

- X – входной алфавит;
- Y – выходной алфавит;
- Q – алфавит внутренних состояний;
- G: $Q\times X\rightarrow Q$ – функция переходов;
- F : $Q\times X\rightarrow Y$ – функция выхода,

называют *автоматом* (автоматом Мили).

Из определения функций G и F следует, что результат их вычисления зависит не только от значения входного символа, но и от внутреннего состояния.

Пусть $G(q, x)=q'$ и $F(q, x)=y$. В этом случае говорят, что под действием входного символа x автомат переходит из состояния q в состояние q', выдавая на выходе символ y. Предполагается, что такое преобразование пары (q, x) в пару (q', y) происходит за один такт условного времени. При этом такты, когда происходит вычисление функций G и F, обозначаются числами натурального ряда t=1, 2, 3, …

Эта специфика вычисления автоматных функций G и F позволяет определить автомат в каноническом виде:

$$\begin{cases} y(t)=F(q(t-1), x(t)), \\ q(t)=G(q(t-1), x(t)), \\ q(0)=q_0, \end{cases}$$

где t=1, 2, …

В этом определении $q(0)=q_0$ – начальное (инициальное) состояние автомата ($q_0\in Q$). Соответственно, $x(t)$, $y(t)$, $q(t)$ являются символами алфавитов X, Y, Q. Параметр t определяет номер такта (t=1, 2, 3,…).

Рис. 7.1. Структура абстрактного автомата

В математике автоматы рассматривают как некие абстрактные устройства (рис. 7.1), которые преобразуют поступающие на вход слова $\xi_x = x(1)\,x(2)x(3)\ldots$ в слова $\xi_y = y(1)\,y(2)y(3)\ldots$

Как видно, структура автомата никак не отражает его внутреннее состояние, за исключением, быть может, начального $q(0)$. Это связано с тем, что внутреннее состояние присуще автомату как «черному ящику», относительно внутренней структуры которого ничего не известно. Определены лишь функции G и F, «вычисляемые» этим абстрактным устройством. Состояние автомата в момент времени t определяется входным символом $x(t)$ и состоянием в предыдущий момент времени $q(t-1)$.

Для знакомства с механизмом преобразования входных слов в выходные предположим, что на вход автомата, который находится в некотором начальном состоянии $q(0)$, поступает входное слово $\xi_x = x(1)\,x(2)x(3)\ldots$ Пару $(\xi_x,\ q)$ называют конфигурацией автомата. Пусть в начальный момент $t = 0$ конфигурация автомата равна $(x(1)x(2)x(3)\ldots,\ q(0))$. Тогда значение выхода и состояния для $t = 1$ определяется как $y(1) = F(q(0),\ x(1))$ и $q(1) = G(q(0),\ x(1))$ соответственно. В результате происходит смена исходной конфигурации на $(x(2)x(3)\ldots,\ q(1))$. Последовательная смена конфигураций для $t = 1,\ 2,\ 3,\ldots$ обеспечивает вычисление последовательности $\xi_y = y(1)\,y(2)y(3)\ldots$ Эти вычисления показывают, что поведение автомата (выходное слово) для заданного входного слова полностью определяется его начальным состоянием.

7.1.1. Классификация автоматов

В зависимости от вида функции выхода автоматы делятся на:

– автоматы *Мили* $\quad F\colon Q \times X \to Y$;

– автоматы *Мура* $F: Q \rightarrow Y$.

В автоматах Мура значение выхода в момент t полностью определяется его состоянием как $y(t)=F(q(t))$.

Автоматы *в зависимости от мощности внутренних состояний* делятся на:

– автоматы *с памятью*, если $k>1$;

– автоматы *без памяти*, если $k=1$.

Функция выхода для автомата без памяти определяется как $y(t)=F(x(t))$, а функция переходов отсутствует.

При теоретических исследованиях *в зависимости от свойств функции перехода G* автоматы подразделяются на:

– *детерминированные* автоматы, если каждой паре (x, q) соответствует не более одного значения $G(x, q)$;

– *недетерминированные* автоматы, когда паре (x, q) соответствует более одного значения $G(x, q)$. В этом случае G является отображением.

В зависимости от мощности входного алфавита X автоматы делятся на:

– *автономные* автоматы, если $|X|=1$;

– автоматы *с входом*, если $|X|>1$.

Поведение автономного автомата полностью определяется его начальным состоянием.

В зависимости от степени определенности функции выхода F автоматы делятся на:

– *полностью определенные* автоматы, если функция F определена для любой пары из $Q \times X$ (для автоматов Мили) либо для любого символа из Q (для автоматов Мура);

– *частично определенные* автоматы, если функция F определена не для всех пар из $Q \times X$ (для автоматов Мили) либо не для всех символов из Q (для автоматов Мура).

В зависимости от структурных свойств алфавитов X, Y, Q автоматы делятся на:

– *абстрактные* автоматы, в которых X, Y, Q задаются произвольными абстрактными символами;

– *структурные* автоматы, в которых X, Y, Q представлены в виде векторов (рис. 7.2);

Рис. 7.2. Структурный автомат

– *смешанные* автоматы, когда некоторые алфавиты являются абстрактными, а некоторые – структурными.

Структурный автомат имеет n входных каналов, m выходных и k внутренних каналов, состояние которых в момент t определяют значения трех векторов: $\overrightarrow{x(t)}$, $\overrightarrow{y(t)}$, $\overrightarrow{q(t)}$. Для структурных автоматов скалярные функции F и G заменяются на векторные:

– $\vec{G} : \vec{Q} \times \vec{X} \rightarrow \vec{Q}$;

– $\vec{F} : \vec{Q} \times \vec{X} \rightarrow \vec{Y}$.

В каноническом виде структурный автомат описывается системой:

$$\begin{cases} \vec{y}(t) = \vec{F}(\vec{q}(t-1), \vec{x}(t)), \\ \vec{q}(t) = \vec{G}(\vec{q}(t-1), \vec{x}(t)), \\ \vec{q}(0) = \vec{q}_0. \end{cases}$$

На практике каждый канал обычно имеет одно из двух состояний, и, как следствие, структурные символы представляются двоичными векторами длины n, m и k соответственно для входов, выходов и внутренних состояний.

В зависимости от свойств функций F и G автоматы делятся на:

– *синхронные* автоматы;

– *асинхронные* автоматы.

Для определения этих классов автоматов вводится понятие *устойчивого состояния*: состояние q автомата устойчиво, если для любой пары (q_i, x), такой, что $G(q_i, x)=q$, справедливо: $G(q, x)=q$. Это означает, что если автомат под действием входного символа x перешел в состояние q, то выйти из этого состояния автомат сможет только тогда, когда произойдет смена входного символа. Автомат считается асинхронным, если все его состояния устойчивы. В противном случае автомат считается синхронным.

Замечание. Введенные понятия синхронности и асинхронности относятся к функциям, а не к устройствам, принцип работы которых также использует эти понятия.

7.1.2. Способы задания автоматов

Способы задания автоматов определяются, прежде всего, способами задания функции выхода F и функции переходов G. Основными из них являются табличный и графический способы.

При табличном способе функции G: $Q \times X \to Q$ и F: $Q \times X \to Y$ задаются двухвходовыми таблицами размерности $|X| \times |Q|$ (табл. 7.1 и табл. 7.2).

Таблица 7.1

Задание функции переходов G: $Q \times X \to Q$

X \ Q	q_1	q_2	…	q_j	…	q_k
x_1	$G(q_1, x_1)$	$G(q_2, x_1)$	…	$G(q_j, x_1)$	…	$G(q_k, x_1)$
x_2	$G(q_1, x_2)$	$G(q_2, x_2)$	…	$G(q_j, x_2)$	…	$G(q_k, x_2)$
…	…	…	…	…	…	…
x_i	$G(q_1, x_i)$	$G(q_2, x_i)$	…	$G(q_j, x_i)$	…	$G(q_k, x_i)$
…	…	…	…	…	…	…
x_n	$G(q_1, x_n)$	$G(q_2, x_n)$	…	$G(q_j, x_n)$	…	$G(q_k, x_n)$

Такое задание функции переходов справедливо как для автоматов Мили, так и для автоматов Мура. Функция выхода для автоматов Мили представлена в табл. 7.2, а для автомата Мура – в табл. 7.3.

Таблица 7.2

Задание функции выхода автомата Мили F: $Q \times X \to Y$

X \ Q	q_1	q_2	…	q_j	…	q_k
x_1	$F(q_1, x_1)$	$F(q_2, x_1)$	…	$F(q_j, x_1)$	…	$F(q_k, x_1)$
x_2	$F(q_1, x_2)$	$F(q_2, x_2)$	…	$F(q_j, x_2)$	…	$F(q_k, x_2)$
…	…	…	…	…	…	…
x_i	$F(q_1, x_i)$	$F(q_2, x_i)$	…	$F(q_j, x_i)$	…	$F(q_k, x_i)$
…	…	…	…	…	…	…
x_n	$F(q_1, x_n)$	$F(q_2, x_n)$	…	$F(q_j, x_n)$	…	$F(q_k, x_n)$

Таблица 7.3

Задание функции выхода автомата Мура F: $Q \to Y$

Q	q_1	q_2	…	q_j	…	q_k
X	$F(q_1)$	$F(q_2)$	…	$F(q_j)$	…	$F(q_k)$

В случае частично определенных функций F и G на пересечении соответствующих строк и столбцов в таблицах ставится прочерк.

При графическом способе автомат задается *графом переходов и выходов*. Для автоматов Мили каждому состоянию $q \in Q$ ставится в соответствие вершина графа переходов и выходов, а дуги соответствуют переходам между ними. Две вершины графа переходов q_i и q_j соединяются дугой, направленной от q_i к q_j, если существует переход $G(q_i, x) = q_j$. Каждой такой дуге сопоставляется пара (x, y), которая определяется значениями $G(q_i, x) = q_j$ и $F(q_i, x) = y$.

$$q_i \xrightarrow{x \qquad y} q_j$$

Для графического задания автоматов Мура применяется несколько другое представление. Каждой вершине графа сопоставляется не только символ внутреннего состояния, но и символ выходного алфавита, определяемый функцией выхода автомата Мура: $F(q)=y$. А дуге, соответствующей переходу из q_i в q_j, приписывается только входной символ x, для которого $G(q_i, x)=q_j$.

На рис. 7.3 приведен пример графического представления автомата $A_{Мили}=\langle X, Y, Q, G, F\rangle$, для которого $X=\{x_1, x_2\}$, $Y=\{y_1, y_2\}$, $Q=\{q_1, q_2, q_3\}$. Табличное представление этого же автомата приведено в табл. 7.4, *а* (функция переходов G) и в табл. 7.4, *б* (функция выхода F).

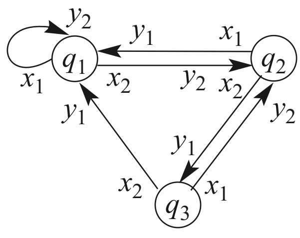


Рис. 7.3. Пример графического представления автомата Мили

Таблица 7.4, а

Функция G: $Q\times X\to Q$ для $A_{Мили}$

X \ Q	q_1	q_2	q_3
x_1	q_1	q_1	q_2
x_2	q_2	q_3	q_1

Таблица 7.4, б

Функция F: $Q\times X\to Y$ для $A_{Мили}$

X \ Q	q_1	q_2	q_3
x_1	y_2	y_1	y_2
x_2	y_2	y_1	y_1

На рис. 7.4 приведен пример графического представления автомата $A_{Мура}=\langle X, Y, Q, G, F\rangle$, для которого $X=\{x_1, x_2\}$, $Y=\{y_1, y_2\}$, $Q=\{q_1, q_2, q_3\}$. Табличное представление этого же автомата приведено в табл. 7.5, *а* (функция переходов G) и в табл. 7.5, *б* (функция выхода F).

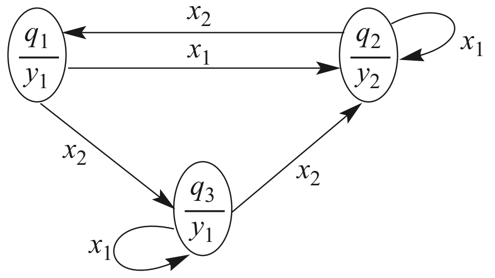


Рис. 7.4. Пример графического представления автомата Мура

Таблица 7.5, а

Функция G: $Q \times X \to Q$ для $\text{А}_{\text{Мура}}$

X \ Q	q_1	q_2	q_3
x_1	q_2	q_2	q_3
x_2	q_3	q_1	q_2

Таблица 7.5, б

Функция F: $Q \times X \to Y$ для $\text{А}_{\text{Мура}}$

Q	q_1	q_2	q_3
X	y_1	y_2	y_1

7.2. Комплексный пример

Рассматриваемый пример связан с построением математической модели устройства управления идеализированным турникетом для пропуска пассажиров в метрополитене. Идеализация связана в первую очередь с дозировкой уровня сложности синтезируемой модели. И делает этот пример обозримым за относительно короткое время.

В качестве математической основы для построения модели инструмента используется понятие конечного автомата со структурным входным и выходным алфавитами.

Предполагается, что турникет – это техническое устройство, которое включает в себя:

– исполнительные органы;
– датчики;
– устройство управления.

Полагается, что к исполнительным органам турникета относятся:

– электромагнит, обеспечивающий открытие створок для прохода пассажира (ЭМ);

– электрическая лампочка (ЭЛ). Для упрощения задачи предполагается, что такая лампочка зеленого цвета одна на турникете, и ее включение означает возможность прохода пассажира;

– электросирена (ЭС), позволяющая привлечь внимание служб метрополитена в случаях, когда имеются многократные (не менее двух) попытки прохода через турникет без оплаты прохода.

На логическом уровне управление исполнительными органами осуществляется по логическим каналам y_1 (управление ЭМ), y_2 (управление ЭЛ) и y_3 (управление ЭС):

$$y_1 = \begin{cases} 1 \text{ – ЭМ включен, проход открыт;} \\ 0 \text{ – ЭМ выключен, проход закрыт;} \end{cases}$$

$$y_2 = \begin{cases} 1 \text{ – ЭЛ включена, приглашение на проход;} \\ 0 \text{ – ЭЛ выключена, проход запрещен;} \end{cases}$$

$$y_3 = \begin{cases} 1 \text{ – ЭС включена;} \\ 0 \text{ – ЭЛ выключена.} \end{cases}$$

Предполагается, что к датчикам относятся:

– датчик фиксации оплаты прохода через турникет (x_1);

– датчик, фиксирующий пассажира в проходе турникета (x_2).

В качестве платежного элемента используются одноразовые магнитные карты либо жетоны, дающие право однократного прохода через турникет. Если карточка была использована, ее повторное использование не оказывает на устройство никакого воздействия.

На логическом уровне управление турникетом происходит на основании входных данных (состояния входных каналов), поступающих по логическому каналу x_1 (от датчика опроса магнитных карт) и по каналу x_2 (от датчика контроля занятости прохода):

$$x_1 = \begin{cases} 1 \text{ – карта обработана (проход оплачен);} \\ 0 \text{ – карта не обработана (проход не оплачен);} \end{cases}$$

$$x_2 = \begin{cases} 1 \text{ – проход занят;} \\ 0 \text{ – проход свободен.} \end{cases}$$

Структура системы управления турникетом приведена на рис. 7.5.

Рис. 7.5. Структура системы управления турникетом

Замечание. Реально обработка магнитных карт (смарт-карт) в турникетах происходит значительно сложнее. Сначала анализируется тип карты на возможность ее обработки, затем проверяется финансовая возможность оплаты, затем происходит запись в соответствующую базу данных измененных значений по карте и еще много других подобных операций. Кроме того, на турникете присутствует не одна, а несколько разноцветных лампочек, которые информируют пассажира о его возможных действиях. Сам проход снабжен целой системой фотодатчиков, которая обеспечивает бесконфликтный проход через турникет добросовестных пассажиров.

По условиям задачи требуется построить математическую модель устройства управления турникетом. Для ее построения необходимо определить функциональные зависимости между входами и выходами с учетом всех возможных ситуаций в процессе работы устройства. Обычно на практике такая исчерпывающая информация содержится в техническом задании.

Исходя из логики работы турникета, устройство управления должно обеспечивать проход очередного пассажира в зависимости от истории, предшествующей его проходу. Работа такого устройства определяется не только значениями входов (входных датчиков), но и его состоянием. И следовательно, конечный автомат является подходящим математическим инструментом для построения такой модели: $A=\langle X, Y, Q, G, F\rangle$.

В качестве входного алфавита X берутся все возможные состояния датчиков x_1 и x_2: $X=\{00, 01, 10, 11\}$. Входной алфавит структурный. Первый элемент пары соответствует состоянию датчика x_1, второй – состоянию датчика x_2.

В качестве выходного алфавита Y следует взять все допустимые управляющие воздействия на исполнительные элементы – y_1, y_2, y_3. К ним можно отнести $Y=\{000, 001, 010, 110\}$. Первый элемент триады соответствует состоянию управления ЭМ (y_1), второй – состоянию управления ЭЛ (y_2), а третий – состоянию управления ЭС (y_3). Отсутствие некоторых структурных символов, таких как 111 в выходном алфавите, связано с их практической недопустимостью. Действительно, недопустимо приглашать пассажира на проход, открыть для прохода створки и одновременно включить тревожную сирену.

Множество состояний Q устройства управления связано с многообразием ситуаций и их последовательностями на входе турникета. Можно выделить три состояния:

– начальное состояние q_1;

– состояние готовности пропуска через турникет одного пассажира (q_2);

– состояние фиксации попытки прохода без оплаты (q_3).

Состояние q_1 является инициальным. В это состояние устройство переходит после пропуска очередного пассажира.

В состояние q_2 устройство переходит при необходимости пропуска одного пассажира. Это возможно только тогда, когда на входе x_1 появляется 1.

В состояние q_3 устройство попадает, если на входе фиксируется символ 01 (попытка прохода через турникет без карты либо с использованной картой). С учетом того, что такая ситуация может возникнуть и без умысла, например по рассеянности пассажира, проход такого пассажира блокируется. Но при повторной попытке прохода через турникет без оплаты кроме блокировки прохода предполагается включение электросирены.

Математическая модель системы управления турникетом в графическом виде приведена на рис. 7.6.

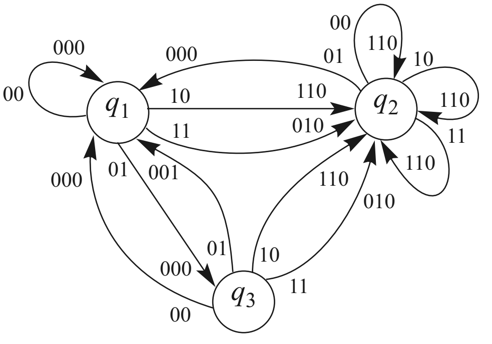


Рис. 7.6. Математическая модель устройства управления турникетом

Исследование модели показывает, что основной цикл работы устройства связан с состояниями q_1 и q_2 и переходами между ними. Переход из q_1 в q_2 по входу 10 соответствует ситуации, когда пассажир прикладывает действующую магнитную карту (жетон) к специальному считывающему устройству. Это приводит к включению электромагнита (y_1=1), в результате чего ограничивающие створки открываются. Одновременно путем установки y_3 в 1 включается ЭЛ, приглашая пассажира на проход через турникет. Переход заканчивается сменой состояния q_1 на q_2.

Переход из состояния q_1 в q_2, как следует из рис. 7.6, возможен также по значению датчиков (11). Предполагается, что такая ситуация на входе возможна у очень «энергичных, стремительных» пассажиров. С учетом того, что метрополитен является транспортным средством повышенной опасности, дверцы для прохода подобных пассажиров будут открываться с запаздыванием, после перехода в состояние q_2.

Переход из состояния q_1 в q_2 соответствует «пустому» такту и происходит по значению (00) входных датчиков с выдачей управляющего воздействия (000).

Переход из состояния q_1 в q_3 соответствует первой по порядку попытке прохода через турникет без оплаты (вход 01). В этом случае выдается управляющий сигнал 000, что соответствует закрытию прохода через турникет.

Состояние q_2, как отмечалось выше, соответствует необходимости пропуска одного пассажира через турникет. Переход реализуется при появлении на входе устройства сигналов 01 либо 11. В первом случае проход пассажира обеспечивает переход устройства в начальное состояние q_1. Во втором случае устройство остается в состоянии q_2 с готовностью пропустить очередного пассажира. Ситуация с входным сигналом (10) в состоянии q_2 требует дополнительных разъяснений. Подобная ситуация возможна в том случае, когда стоящий перед турникетом многократно прикладывает магнитную карту (жетоны) к считывающему устройству без прохода через турникет. Как устройство должно реагировать на подобные действия? Решения могут быть разнообразными, но в данном примере принято следующее. Вне зависимости от числа появления на входе устройства сигнала (10) через устройство сможет пройти только один пассажир.

Замечание. Возможны и другие варианты разрешения описанной выше ситуации. Возможно «запоминание» числа появлений на входе сигналов (10), что приведет к росту числа состояний. Но при этом число состояний должно быть ограничено некоторой конкретной величиной. Возможно встраивание управления устройством считывания магнитных карт (жетонов), не допускающее преждевременное считывание (ввод) до перехода устройства в начальное состояние.

Состояние q_3 соответствует однократной попытке прохода через турникет без оплаты. Если в этом состоянии вновь на выходе появляется сигнал (01), то выдается управляющее воздействие (001), которое приводит к включению электросирены. Устройство переходит в начальное состояние. Если же на входе появляется сигнал (00), устройство переходит в начальное состояние. Переходы из q_3 по сигналам 10, 11 полностью совпадают с переходами из q_1.

После определения математической модели устройства следует уточнить выходной алфавит Y. Анализ показывает, что $Y=\{000, 001, 010, 110\}$. Все остальные триады недопустимы.

Читателю в качестве упражнения предлагается получить табличное представление функций G, F.

7.3. Эквивалентирование автоматов Мили и Мура

Одним из вопросов, который рассматривается в теории автоматов, является вопрос о соотношении класса автоматов Мили и класса автоматов Мура. Для получения ответа на этот вопрос необходимо определить отношение эквивалентности на множестве автоматов.

Считается, что автомат $A^1=\langle X^1, Y^1, Q^1, G^1, F^1\rangle$ является подавтоматом автомата $A^2=\langle X^2, Y^2, Q^2, G^2, F^2\rangle$ ($A^1\subseteq A^2$), если:

– входной алфавит A^1 является подмножеством входного алфавита A^2: $X^1\subseteq X^2$;

– выходной алфавит A^1 является подмножеством выходного алфавита A^2: $Y^1\subseteq Y^2$;

– для каждого состояния q^1 и любого входного слова $\xi_x=x(1)x(2)x(3)\dots x(\mathrm{s})$ автомата A^1 найдется состояние q^2 автомата A^2, такое, что выполняется равенство $F^1(q^1, \xi_x)=F^2(q^2, \xi_x)$.

Два автомата A^1 и A^2 эквивалентны, если $A^1\subseteq A^2$ и $A^2\subseteq A^1$.

В теории автоматов известна следующая теорема.

<u>ТЕОРЕМА</u>. Класс автоматов Мили и класс автоматов Мура – равномощны.

Из этой теоремы непосредственно следует, что для каждого автомата Мили существует эквивалентный ему автомат Мура. И обратно, для каждого автомата Мура существует эквивалентный ему автомат Мили.

Доказательством этого утверждения может служить процедура преобразования автомата Мили в эквивалентный ему автомат Мура и процедура преобразования автомата Мура в эквивалентный ему автомат Мили.

<u>Задача 1</u>. *Дано*: автомат Мура $A^1=\langle X, Y, Q^1, G^1, F^1\rangle$.

Определить (найти): автомат Мили $A=\langle X, Y, Q, G, F\rangle$, эквивалентный A^1.

На рис. 7.7 иллюстрируется процедура преобразования автомата Мура при его графическом представлении. Слева показан фрагмент автомата Мура: переход из состояния q_i^1 в состояние q_j^1. Преобразование этого фрагмента в фрагмент эквивалентного ему автомата Мили заключается в переносе из состояния q_j^1 выходного символа y на все входящие в него стрелки (переходы). Справа на рисунке показан результирующий фрагмент автомата Мили.

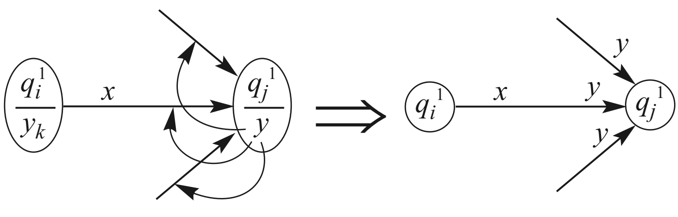


Рис. 7.7. Преобразование автомата Мура в автомат Мили

В терминах функций G^1, F^1 соответствующие функции G, F эквивалентного автомата Мили определяются следующим образом:

– $Q=Q^1$;

– $G(q_i, x)=G^1(q_i, x)$;

– $F(q_i, x)=F^1(G^1(q_i, x))$.

<u>Задача 2</u>. *Дано*: автомат Мили $A^1=\langle X, Y, Q, G^1, F^1\rangle$.

Определить (найти): автомат Мура $A=\langle X, Y, S, G, F\rangle$, эквивалентный A^1.

На рис. 7.8 иллюстрируется процедура преобразования автомата Мили при его графическом представлении. Преобразование этого фрагмента во фрагмент эквивалентного ему автомата Мура заключается в «расщеплении» состояния q_j^1 на несколько новых состояний с добавлением к каждому из них выходного символа. Справа на рис. 7.8 показан результирующий фрагмент автомата Мура.

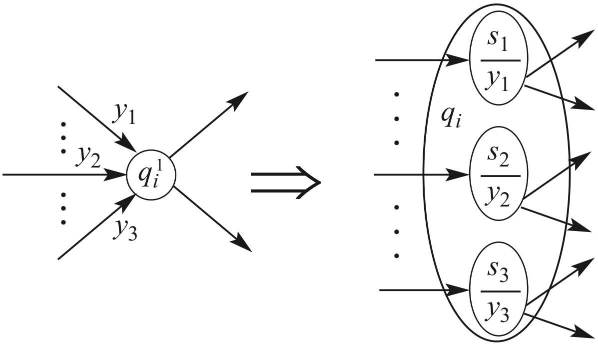


Рис. 7.8. Преобразование автомата Мили в автомат Мура

Преобразование заключается в расщеплении состояния q_i автомата Мили на ряд состояний автомата Мура s_1, s_2 и т.д. для каждого выходного символа на переходах q_i^1. Каждому состоянию s_1, s_2 и т.д. автомата Мура ставится в соответствие выходной символ, переносимый с входящей стрелки. В терминах функций G^1, F^1 соответствующие функции G, F эквивалентного автомата Мура и множество состояний S определяются следующим:

– $S=Q^1\times Y=\{s=(q, y)|q\in Q^1, y\in Y\}$;

– $F(s)=F(q, y)=y$;

– $G(s, x)=G((q, y), x)=(G^1(q, x), F^1(q, x))=(q', y')=s'$.

Таким образом, преобразование автомата Мили в эквивалентный автомат Мура может приводить к увеличению числа состояний.

Пример задачи преобразования автомата Мура в автомат Мили.

Дано: автомат Мура $A^1=\langle X, Y, Q^1, G^1, F^1\rangle$, заданный в табличном виде, функции переходов и выходов для которого приведены в табл. 7.6, *а* и в табл. 7.6, *б*, соответственно.

Таблица 7.6, а

Функция G^1: $Q\times X\to Q$ для A^1

X \ Q	q_1	q_2	q_3
x_1	q_2	q_1	q_2
x_2	q_3	q_2	q_1

Таблица 7.6, б

Функция F^1: $Q\times X\to Y$ для A^1

Q	q_1	q_2	q_3
Y	y_1	y_2	y_2

Определить (найти): автомат Мили $A=\langle X, Y, Q, G, F\rangle$, эквивалентный заданному автомату Мура.

Решение. Согласно описанной выше процедуре преобразования

$Q=Q^1=\{q_1, q_2, q_3\}$;

$G=G^1$, и, следовательно, табл. 7.6, *а* задает функцию переходов искомого автомата.

Функция выхода искомого автомата $F(q, x)=F^1(G^1(q, x))$:

$F(q_1, x_1)=F^1(G^1(q_1, x_1))=F^1(q_2)=y_2$;

$F(q_1, x_2)=F^1(G^1(q_1, x_2))=F^1(q_3)=y_2$;

$F(q_2, x_1)=F^1(G^1(q_2, x_1))=F^1(q_1)=y_1$;

$F(q_2, x_2)=F^1(G^1(q_2, x_2))=F^1(q_2)=y_2$;

$F(q_3, x_1)=F^1(G^1(q_3, x_1))=F^1(q_2)=y_2$;

$F(q_3, x_3)=F^1(G^1(q_3, x_3))=F^1(q_1)=y_1$.

В табл. 7.6, *в* приведена найденная функция выходов автомата искомого автомата Мили A^2.

Таблица 7.6, в

Функция F: $Q \times X \to Y$ для А

$X \backslash Q$	q_1	q_2	q_3
x_1	y_2	y_1	y_2
x_2	y_2	y_2	y_1

Пример задачи преобразования автомата Мили в автомат Мура.

Дано: автомат Мили $A^1 = \langle X, Y, Q^1, G^1, F^1 \rangle$, заданный в табличном виде, функции переходов и выходов которого приведены в табл. 7.7, *а* и в табл. 7.7, *б* соответственно.

Таблица 7.7, а

Функция G^1: $Q \times X \to Q$ для A^1

$X \backslash Q$	q_1	q_2	q_3
x_1	q_2	q_2	q_1
x_2	q_3	q_3	q_2

Таблица 7.7, б

Функция F^1: $Q \times X \to Y$ для A^1

$X \backslash Q$	q_1	q_2	q_3
x_1	y_1	y_2	y_2
x_2	y_1	y_1	y_1

Определить (найти): автомат Мура $A = \langle X, Y, S, G, F \rangle$, эквивалентный заданному автомату Мили.

Решение. Согласно описанной выше процедуре преобразования

$S = Q^1 \times Y = \{q_1, q_2, q_3\} \times \{y_1, y_2\} = \{s_1 = (q_1, y_1), s_2 = (q_1, y_2), s_3 = (q_2, y_1), s_4 = (q_2, y_2), s_5 = (q_3, y_1), s_6 = (q_3, y_2)\}$.

Функция выхода искомого автомата F: $S \to Y$:

$F(s_1) = F((q_1, y_1)) = y_1$;

$F(s_2) = F((q_1, y_2)) = y_2$;

$F(s_3) = F((q_2, y_1)) = y_1$;

$F(s_4) = F((q_2, y_2)) = y_2$;

$F(s_5) = F((q_3, y_1)) = y_1$;

$F(s_6) = F((q_3, y_2)) = y_2$.

Функция переходов искомого автомата G: $S \times X \to Y$:

$G(s_1, x_1) = G((q_1, y_1), x_1) = (G^1(q_1, x_1), F^1(q_1, x_1)) = (q_2, y_1) = s_3$;

$G(s_1, x_2) = G((q_1, y_1), x_2) = (G^1(q_1, x_2), F^1(q_1, x_2)) = (q_2, y_1) = s_5$;

$G(s_2, x_1) = G((q_1, y_2), x_1) = (G^1(q_1, x_1), F^1(q_1, x_1)) = (q_2, y_2) = s_3$;

$G(s_2, x_2) = G((q_1, y_2), x_1) = (G^1(q_1, x_2), F^1(q_1, x_2)) = (q_3, y_1) = s_5$;

$G(s_3, x_1) = G((q_2, y_1), x_1) = (G^1(q_2, x_1), F^1(q_2, x_1)) = (q_2, y_2) = s_4$;

$G(s_3, x_2) = G((q_2, y_1), x_1) = (G^1(q_2, x_2), F^1(q_2, x_2)) = (q_3, y_1) = s_5$;

$G(s_4, x_1) = G((q_2, y_2), x_1) = (G^1(q_2, x_1), F^1(q_2, x_1)) = (q_2, y_2) = s_4$;

$G(s_4, x_2) = G((q_2, y_2), x_1) = (G^1(q_2, x_2), F^1(q_2, x_2)) = (q_3, y_1) = s_5$;

$G(s_5, x_1) = G((q_3, y_1), x_1) = (G^1(q_3, x_1), F^1(q_3, x_1)) = (q_1, y_2) = s_2$;

$G(s_5, x_2) = G((q_3, y_1), x_1) = (G^1(q_3, x_2), F^1(q_3, x_2)) = (q_2, y_1) = s_3$;

$G(s_6, x_1) = G((q_3, y_2), x_1) = (G^1(q_3, x_1), F^1(q_3, x_1)) = (q_1, y_2) = s_2$;

$G(s_6, x_2) = G((q_3, y_2), x_1) = (G^1(q_3, x_2), F^1(q_3, x_2)) = (q_2, y_1) = s_3$.

В табл. 7.8, *а* и в табл. 7.8, *б* приведены полученные функции переходов и выхода для искомого автомата Мура.

Таблица 7.8, а

Функция G: $S \times X \to S$ для A

X \ S	s_1	s_2	s_3	s_4	s_5	s_6
x_1	s_3	s_3	s_4	s_4	s_2	s_2
x_2	s_5	s_5	s_5	s_5	s_3	s_3

Таблица 7.8, б

Функция F: $S \to Y$ для A

S	s_1	s_2	s_3	s_4	s_5	s_6
Y	y_1	y_2	y_1	y_2	y_1	y_2

7.4. Минимизация конечных автоматов

Пусть А=$\langle X, Y, Q, G, F\rangle$ – некоторый автомат. Два состояния q_i и q_j автомата А эквивалентны ($q_i \sim q_j$), если равенство $F(q_i, \xi_x)=F(q_j, \xi_x)$ выполняется для любого входного слова $\xi_x = x(1)x(2)x(3)\ldots x(\mathrm{s})$. Автомат считается *минимальным* по числу состояний, если из эквивалентности состояний q_i и q_j следует, что $q_i = q_j$. Иными словами, автомат считается минимальным, если в нем нет ни одной пары эквивалентных состояний.

Эквивалентные неравные между собой состояния в автомате можно «стянуть» в одно. С учетом этого поиск минимального автомата сводится к выполнению следующих этапов:

– Выделение классов (подмножеств) эквивалентных между собой состояний. Число полученных классов определяет число состояний минимального автомата.

– Построение функций переходов и выхода минимального автомата «стягиванием» всех состояний класса в одно обобщенное состояние.

Для выделения классов эквивалентных состояний вводится более слабое понятие эквивалентности состояний: k-эквивалентные состояния, где k – натуральное число ($k=1, 2, \ldots$).

Два состояния q_i и q_j k-эквивалентны ($q_i \overset{k}{\sim} q_j$), если равенство $F(q_i, \xi_x^k)=F(q_j, \xi_x^k)$ выполняется на всех входных последовательностях ξ_x^k, по длине не превосходящих k. Для каждого k ($k=1, 2, \ldots$) k-эквивалентность определяет разбиение множества состояний на классы k-эквивалентных состояний. В силу определения классы k-эквивалентности либо совпадают с классами ($k-1$)-эквивалентности, либо образуют разбиение некоторого класса ($k-1$)-эквивалентности.

Алгоритм выделения классов эквивалентных состояний для автомата Мили

Начальный шаг. Для $k=1$ находятся 1-эквивалентные состояния посредством испытания автомата односимвольными входными последовательностями. Число таких последовательностей совпадает с мощностью входного алфавита, а сами классы выделяются с помощью функции выхода по совпадению столбцов при ее табличном задании. Пусть $\{K_{11}, K_{12}, \ldots, K_{1,\, \mathrm{n}1}\}$ – классы 1-эквивалентности.

Итерационный (k-й) шаг ($k>1$). Пусть на $k-1$ шаге были получены $\{K_{k-1,\,1}, K_{k-1,\,2}, \ldots, K_{k\text{-}1,\,p}\}$ классы $k-1$ эквивалентности. Для определения классов k-эквивалентности строится модифицированная функция переходов G: $Q\times X \rightarrow \{K_{k-1,\,1}, K_{k-1,\,2}, \ldots, K_{k\text{-}1,\,p}\}$, где вместо $G(q, x)$ подставляется K_j из $\{K_{k-1,\,1}, K_{k-1,\,2}, \ldots, K_{k\text{-}1,\,p}\}$, для которого выполняется условие $G(q, x)\in K_j$. По совпадению столбцов внутри классов $\{K_{k-1,\,1}, K_{k-1,\,2}, \ldots, K_{k\text{-}1,\,p}\}$ эквивалентности, найденных на предыдущем шаге, находятся классы k-эквивалентности.

Правило остановки. Работа алгоритма прекращается, когда на очередном шаге происходит повторение результатов (классов), найденных на предыдущем шаге.

Пусть $S=\{s_1, s_2, \ldots, s_m\}$ – разбиение множества состояний на m-классов эквивалентности. Для этих классов справедливо:

$s_i \subseteq Q$;

$\bigcup\limits_{i=1\ldots m} s_i = Q$;

$s_i \cap s_j = \varnothing$, если $i \neq j$.

Задача нахождения минимального автомата $\mathrm{A}_{\text{мин}}=\langle X, Y, S, \widetilde{G}, \widetilde{F}\rangle$ сводится к построению функции переходов $\widetilde{G}$: $S\times X\rightarrow S$ и функции выхода $\widetilde{F}$: $S\times X\rightarrow Y$ для заданного автомата $\mathrm{A}=\langle X, Y, Q, G, F\rangle$.

<u>Пример</u>.

Дано: автомат Мили $A=\langle X, Y, Q, G, F\rangle$, заданный в табличном виде, функции переходов и выходов которого приведены в табл. 7.9, *а* и в табл. 7.9, *б* соответственно.

Определить (найти): минимальный автомат Мили $A_{мин}=\langle X, Y, S, \widetilde{G}, \widetilde{F}\rangle$, эквивалентный заданному автомату.

<u>Решение</u>.

Согласно описанному выше алгоритму.

Начальный шаг. Для $k=1$ находятся 1-эквивалентные состояния по совпадению столбцов в функции выхода.

$K_1=\{K_{11}, K_{12}\}$, $K_{11}=\{q_1, q_2, q_5\}$, $K_{12}=\{q_3, q_4, q_6, q_7\}$.

Итерационный шаг ($k=2$).

Модифицированная функция переходов G: $Q\times X\to K_1$ приведена в табл. 7.9, *в*.

Классы 2-эквивалентных состояний находятся внутри найденных на предыдущем шаге классов: $K_2=\{K_{21}, K_{22}, K_{23}\}$, $K_{21}=\{q_1, q_2, q_5\}$, $K_{22}=\{q_3, q_6\}$, $K_{23}=\{q_4, q_7\}$.

Итерационный шаг ($k=3$).

Модифицированная функция переходов G: $Q\times X\to K_2$ приведена в табл. 7.9, *г*.

Классы 3-эквивалентных состояний находятся внутри найденных на предыдущем шаге классов: $K_3=\{K_{31}, K_{32}, K_{33}\}$, $K_{31}=\{q_1, q_2, q_5\}$, $K_{32}=\{q_3, q_6\}$, $K_{33}=\{q_4, q_7\}$. Найденное решение повторяет решение, полученное на предыдущем шаге, и поиск классов k-эквивалентности прекращается. Следовательно, минимальный автомат содержит три состояния $S=\{s_1, s_2, s_3,\}$, для которых $s_1=\{q_1, q_2, q_5\}$, $s_2=\{q_3, q_6\}$, $s_3=\{q_4, q_7\}$.

Построение минимального автомата $A_{мин}=\langle X, Y, S, \widetilde{G}, \widetilde{F}\rangle$.

В табл. 7.10, *а* и в табл. 7.10, *б* приведены функции $\widetilde{G}$ и $\widetilde{F}$ минимального автомата.

Таблица 7.9, а

Функция G: $Q \times X \to Q$ для A

$X \setminus Q$	q_1	q_2	q_3	q_4	q_5	q_6	q_7
x_1	q_2	q_5	q_7	q_1	q_5	q_4	q_5
x_2	q_3	q_6	q_4	q_3	q_6	q_7	q_6

Таблица 7.9, б

Функция F: $Q \times X \to Y$ для A

$X \setminus Q$	q_1	q_2	q_3	q_4	q_5	q_6	q_7
x_1	y_1	y_1	y_2	y_2	y_1	y_2	y_2
x_2	y_2	y_2	y_1	y_1	y_2	y_1	y_1

Таблица 7.9, в

Функция G: $Q \times X \to K_1$ для A

$X \setminus Q$	q_1	q_2	q_3	q_4	q_5	q_6	q_7
x_1	K_{11}	K_{11}	K_{12}	K_{11}	K_{11}	K_{12}	K_{11}
x_2	K_{12}	K_{12}	K_{12}	K_{12}	K_{12}	K_{12}	K_{12}

Таблица 7.9, г

Функция G: $Q \times X \to K_2$ для A

$X \setminus Q$	q_1	q_2	q_3	q_4	q_5	q_6	q_7
x_1	K_{21}	K_{21}	K_{23}	K_{21}	K_{21}	K_{23}	K_{21}
x_2	K_{22}	K_{22}	K_{23}	K_{22}	K_{22}	K_{23}	K_{22}

Таблица 7.10, а

Функция $\widetilde{G}$: $S \times X \to S$

$X \setminus S$	s_1	s_2	s_3
x_1	s_1	s_3	s_1
x_2	s_2	s_3	s_2

Таблица 7.10, б

Функция $\widetilde{F}$: $S \times X \to Y$

$X \setminus S$	s_1	s_2	s_3
x_1	y_1	y_2	y_2
x_2	y_2	y_1	y_1

7.5. Операции над автоматами

Для автоматов как для математических объектов определен ряд операций, которые позволяют из автоматов строить новые автоматы. Эти операции называют операциями композиции. К ним относятся:

– операция *параллельной композиции*;
– операция *последовательной композиции*;
– операция *композиции с обратной связью*.

Операция параллельной композиции

Пусть $A^1=\langle X, Y^1, Q^1, G^1, F^1\rangle$ и $A^2=\langle X, Y^2, Q^2, G^2, F^2\rangle$ – два автомата, для определенности, два автомата Мили, а φ: $Y^1\times Y^2\to Y$ – преобразователь без памяти.

Операция параллельной композиции автоматов A^1 и A^2 на структурном уровне показана на рис. 7.9.

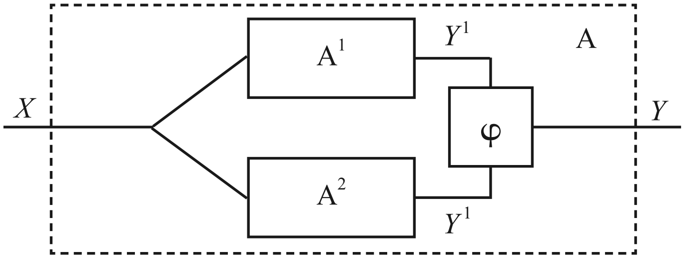


Рис. 7.9. Параллельная композиция

Пусть $A=\langle X, Y, S, G, F\rangle$ – результирующий автомат. Его границы на рис. 7.9 обведены пунктиром. Функции G и F описывают поведение композиции как единого целого.

Множество состояний результирующего автомата S есть декартово произведение состояний автомата A^1 и автомата A^2: $S=Q^1\times Q^2=\{s_i=(q_i^1,\ q_i^2)\,|\,q_i^1\in Q^1,\ q_i^2\in Q^2\}$. Таким образом, каждое состояние результирующего автомата представляется упорядоченной парой состояний компонентных автоматов.

Входной алфавит результирующего автомата совпадает с входным алфавитом для автоматов A^1 и A^2.

Выходной алфавит определяется множеством возможных значений функции φ: $Y^1 \times Y^2 \to Y$ и совпадает с Y.

Функция переходов G: $S \times X \to S$ определяется следующим образом:

$G(s, x) = G((q^1, q^2), x) = (G^1(q^1, x), G^2(q^2, x)) = \widetilde{s}$, где $s = (q^1, q^2)$, $\widetilde{s}$ – состояние, в которое переходит результирующий автомат из состояния s под действием входного символа x.

Состояние $\widetilde{s}$ определяется как упорядоченная пара $\widetilde{s} = (\widetilde{q}^1, \widetilde{q}^2)$, каждый из элементов которой есть результат вычисления соответствующей функции переходов компонентных автоматов:

$\widetilde{q}^1 = G^1(q^1, x)$;

$\widetilde{q}^2 = G^2(q^1, x)$.

Функция выхода F: $S \times X \to Y$ определяется следующим образом:

$F(s, x) = \varphi(F^1(q^1, x), F^2(q^2, x)) = \varphi(y^1, y^1) = y$,
где $s = (q^1, q^2)$, $y_1 = F^1(q^1, x)$, $y^2 = F^2(q^2, x)$.

Ниже приведен пример параллельной композиции двух автоматов.

Пример.

Дано: автомат Мили $A^1 = \langle X, Y^1, Q^1, G^1, F^1 \rangle$ (табл. 7.11, *а* и табл. 7.11, *б*), автомат Мили $A^2 = \langle X, Y^2, Q^2, G^2, F^2 \rangle$ (табл. 7.12, *а* и табл. 7.12, *б*) и функция φ: $Y^1 \times Y^2 \to Y$ (табл. 7.13).

Определить (найти): автомат $A = \langle X, Y, S, G, F \rangle$, который является результатом параллельной композиции исходных автоматов (рис. 7.9).

Таблица 7.11, а

Функция G^1

$X \backslash Q^1$	q_{11}	q_{12}
x_1	q_{11}	q_{11}
x_2	q_{12}	q_{11}

Таблица 7.11, б

Функция F^1

$X \backslash Q^1$	q_{11}	q_{12}
x_1	y_{11}	y_{13}
x_2	y_{12}	y_{12}

Таблица 7.12, а

Функция G^2

$X \backslash Q^2$	q_{21}	q_{22}	q_{23}
x_1	q_{22}	q_{23}	q_{21}
x_2	q_{23}	q_{21}	q_{22}

Таблица 7.12, б

Функция F^2

$X \backslash Q^2$	q_{21}	q_{22}	q_{23}
x_1	y_{21}	y_{22}	y_{21}
x_2	y_{22}	y_{21}	y_{22}

Таблица 7.13

Функция φ

$Y^1 \backslash Y^2$	y_{21}	y_{22}
y_{11}	y_1	y_2
y_{12}	y_3	y_1
y_{13}	y_2	y_3

<u>Решение</u>.

$X=\{x_1, x_2\}$ – входной алфавит композиции.

$Y=\{y_1, y_2, y_3\}$ – выходной алфавит композиции.

$S=Q^1 \times Q^2=\{q_{11}, q_{12}\} \times \{q_{21}, q_{22}, q_{23}\}=\{s_1=(q_{11}, q_{21}), s_2=(q_{11}, q_{22}), s_3=(q_{11}, q_{23}), s_4=(q_{12}, q_{21}), s_5=(q_{12}, q_{22}), s_6=(q_{12}, q_{23})\}$.

Функция переходов G: $S \times X \to S$.

$G(s_1, x_1)=G((q_{11}, q_{21}), x_1)=(G^1(q_{11}, x_1), G^2(q_{21}, x_1))=(q_{11}, q_{22})=s_2$;

$G(s_1, x_2)=G((q_{11}, q_{21}), x_2)=(G^1(q_{11}, x_2), G^2(q_{21}, x_2))=(q_{12}, q_{23})=s_6$;

$G(s_2, x_1)=G((q_{11}, q_{22}), x_1)=(G^1(q_{11}, x_1), G^2(q_{22}, x_1))=(q_{11}, q_{23})=s_3$;

$G(s_2, x_2)=G((q_{11}, q_{22}), x_2)=(G^1(q_{11}, x_2), G^2(q_{22}, x_2))=(q_{12}, q_{21})=s_4$;

$G(s_3, x_1)=G((q_{11}, q_{23}), x_1)=(G^1(q_{11}, x_1), G^2(q_{23}, x_1))=(q_{11}, q_{21})=s_1$;

$G(s_3, x_2)=G((q_{11}, q_{23}), x_2)=(G^1(q_{11}, x_2), G^2(q_{23}, x_2))=(q_{12}, q_{22})=s_5$;

$G(s_4, x_1)=G((q_{12}, q_{21}), x_1)=(G^1(q_{12}, x_1), G^2(q_{21}, x_1))=(q_{11}, q_{22})=s_2$;

$G(s_4, x_2)=G((q_{12}, q_{21}), x_2)=(G^1(q_{12}, x_2), G^2(q_{21}, x_2))=(q_{11}, q_{23})=s_3$;

$G(s_5, x_1)=G((q_{12}, q_{22}), x_1)=(G^1(q_{12}, x_1), G^2(q_{22}, x_1))=(q_{11}, q_{23})=s_2$;

$G(s_5, x_2)=G((q_{12}, q_{22}), x_2)=(G^1(q_{12}, x_2), G^2(q_{22}, x_2))=(q_{11}, q_{21})=s_1$;

$G(s_6, x_1) = G((q_{12}, q_{23}), x_1) = (G^1(q_{12}, x_1), G^2(q_{23}, x_1)) = (q_{11}, q_{21}) = s_1$;

$G(s_6, x_2) = G((q_{12}, q_{23}), x_2) = (G^1(q_{12}, x_2), G^2(q_{23}, x_2)) = (q_{11}, q_{22}) = s_2$.

– Функция выхода F: $S \times X \rightarrow Y$.

$F(s_1, x_1) = F((q_{11}, q_{21}), x_1) = \varphi(F^1(q_{11}, x_1), F^2(q_{21}, x_1)) = \varphi(y_{11}, y_{21}) = y_1$;

$F(s_1, x_2) = F((q_{11}, q_{21}), x_2) = \varphi(F^1(q_{11}, x_2), F^2(q_{21}, x_2)) = \varphi(y_{12}, y_{22}) = y_1$;

$F(s_2, x_1) = F((q_{11}, q_{22}), x_1) = \varphi(F^1(q_{11}, x_1), F^2(q_{22}, x_1)) = \varphi(y_{11}, y_{22}) = y_2$;

$F(s_2, x_2) = F((q_{11}, q_{22}), x_2) = \varphi(F^1(q_{11}, x_2), F^2(q_{22}, x_2)) = \varphi(y_{12}, y_{21}) = y_3$;

$F(s_3, x_1) = F((q_{11}, q_{23}), x_1) = \varphi(F^1(q_{11}, x_1), F^2(q_{23}, x_1)) = \varphi(y_{11}, y_{21}) = y_1$;

$F(s_3, x_2) = F((q_{11}, q_{23}), x_2) = \varphi(F^1(q_{11}, x_2), F_2(q_{23}, x_2)) = \varphi(y_{12}, y_{22}) = y_1$;

$F(s_4, x_1) = F((q_{12}, q_{21}), x_1) = \varphi(F^1(q_{12}, x_1), F_2(q_{21}, x_1)) = \varphi(y_{13}, y_{21}) = y_2$;

$F(s_4, x_2) = F((q_{12}, q_{21}), x_2) = \varphi(F^1(q_{12}, x_2), F_2(q_{21}, x_2)) = \varphi(y_{12}, y_{22}) = y_1$;

$F(s_5, x_1) = F((q_{12}, q_{22}), x_1) = \varphi(F^1(q_{12}, x_1), F_2(q_{22}, x_1)) = \varphi(y_{13}, y_{22}) = y_3$;

$F(s_5, x_2) = F((q_{12}, q_{22}), x_2) = \varphi(F^1(q_{12}, x_2), F^2(q_{22}, x_2)) = \varphi(y_{12}, y_{21}) = y_3$;

$F(s_6, x_1) = F((q_{12}, q_{23}), x_1) = \varphi(F^1(q_{12}, x_1), F^2(q_{23}, x_1)) = \varphi(y_{13}, y_{21}) = y_2$;

$F(s_6, x_2) = F((q_{12}, q_{23}), x_2) = \varphi(F^1(q_{12}, x_2), F^2(q_{23}, x_2)) = \varphi(y_{12}, y_{22}) = y_1$.

В табл. 7.14, *а* и табл. 7.14, *б* приведены функция переходов и функция выхода результирующего автомата.

Таблица 7.14, а

Функция переходов G: $S \times X \rightarrow S$

$X \backslash S$	s_1	s_2	s_3	s_4	s_5	s_6
x_1	s_2	s_3	s_1	s_2	s_2	s_1
x_2	s_6	s_4	s_5	s_3	s_1	s_2

Таблица 7.14, б

Функция выхода F: $S \times X \rightarrow Y$

$X \backslash S$	s_1	s_2	s_3	s_4	s_5	s_6
x_1	y_1	y_2	y_1	y_2	y_3	y_2
x_2	y_1	y_3	y_1	y_1	y_3	y_1

Операция последовательной композиции

Пусть $A^1=\langle X, Z, Q^1, G^1, F^1\rangle$ и $A^2=\langle Z, Y, Q^2, G^2, F^2\rangle$ – два автомата, для определенности, два автомата Мили.

Операция последовательной композиции автоматов A^1 и A^2 на структурном уровне показана на рис. 7.10.

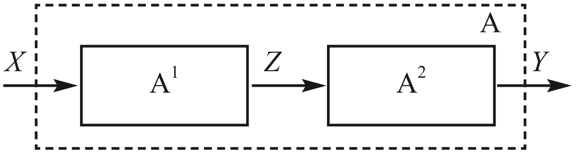


Рис. 7.10. Последовательная композиция

Пусть $A=\langle X, Z, S, G, F\rangle$ – результирующий автомат. Его границы на рис. 7.10 обведены пунктиром. Функции G и F описывают поведение двух автоматов как единого целого.

Множество состояний результирующего автомата S есть декартово произведение состояний автомата A^1 и автомата A^2: $S=Q^1\times Q^2=\{s_i=(q_i^1, q_i^2)|q_i^1\in Q^1, q_i^2\in Q^2\}$. Таким образом, каждое состояние результирующего автомата представляется упорядоченной парой состояний компонентных автоматов.

Входной алфавит результирующего автомата совпадает с входным алфавитом для автомата A^1.

Выходной алфавит результирующего автомата совпадает с выходным алфавитом автомата A^2.

Функция выхода F: $S\times X\to Y$ определяется следующим образом:

$F(s, x)=F((q^1, q^2), x)=F^2(q^2, F_1(q^1, x))=F^2(q^2, z)=y$,

где $s=(q^1, q^2)$ – исходное состояние композиции;

$z=F^1(q^1, x)$ – значение выхода автомата A^1;

$y=F^2(q^2, z)$ – значение выхода автомата A^2, которое является выходом композиции.

Функция переходов G: $S\times X\to S$ определяется следующим образом:

$$G(s, x) = G((q^1, q^2, x) = (G^1(q^1, x), G^2(q^2, F^1(q^1, x))) = \tilde{s},$$

где $\tilde{s}$ – состояние, в которое переходит результирующий автомат из состояния s под действием входного символа x. Состояние $\tilde{s}$ определяется как упорядоченная пара $\tilde{s} = (\tilde{q}^1, \tilde{q}^2)$, каждый из элементов которой есть результат вычисления соответствующей функции переходов компонентных автоматов:

$$\tilde{q}^1 = G^1(q^1, x);$$

$$\tilde{q}^2 = G^2(q^2, F^1(q^1, x)).$$

Необходимо отметить, что вычисление функции переходов $G(s, x)$ для последовательной композиции происходит за два условных такта. На первом такте выполняется вычисление функций F^1 и G^1 автомата A^1. На втором такте выполняется вычисление функций F^2 и G^2 автомата A^2. На практике такое соединение приводит к снижению быстродействия.

Операция композиции с обратной связью

Пусть $A^1 = \langle Z, Y, Q^1, G^1, F^1 \rangle$ и $A^2 = \langle Y, V, Q^2, G^2, F^2 \rangle$ – два автомата, а φ: $X \times V \rightarrow Z$ – преобразователь без памяти.

Операция композиции автоматов A^1 и A^2 с обратной связью на структурном уровне показана на рис. 7.11.

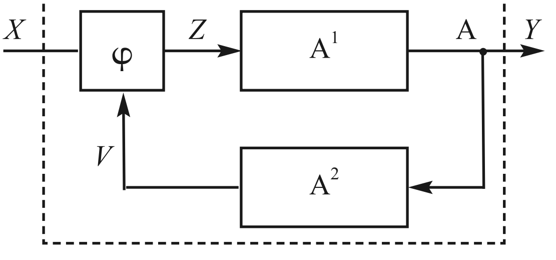


Рис. 7.11. Композиция с обратной связью

Композиция будет автоматом, если, по крайней мере, один из автоматов A^1 и A^2 является автоматом Мура. Это требование связано с обеспечением «устойчивости» при вычислении поведения системы автоматов как единого целого.

Пусть $A = \langle Z, Y, S, G, F \rangle$ – результирующий автомат. Его границы на рис. 7.11 обведены пунктиром. Функции G и F описывают поведение композиции как единого целого. Для определенности, при выводе функций G и F полагается, что A^1 – автомат Мили, а A^2 – автомат Мура.

Множество состояний результирующего автомата S есть декартово произведение состояний автомата A^1 и автомата A^2:

$$S = Q^1 \times Q^2 = \{s_i = (q_i^1, q_i^2) | q_i^1 \in Q^1, q_i^2 \in Q^2\}.$$

Таким образом, каждое состояние результирующего автомата представляется упорядоченной парой состояний компонентных автоматов.

Входной алфавит результирующего автомата совпадает с входным алфавитом X.

Выходной алфавит совпадает с алфавитом Y.

Функция выхода F: $S \times X \rightarrow Y$ определяется следующим образом:

$$F(s, x) = F((q^1, q^2), x) = F^1(q^1, z) = F^1(q^1, \varphi(x, v)) = F^1(q^1, \varphi(x, F^2(q^2)) = y,$$

где $s = (q^1, q^2)$;

$z = \varphi(x, v)$;

$v = F^2(q^2)$.

Функция переходов G: $S \times X \rightarrow S$ определяется следующим образом:

$$G(s, x) = G((q^1, q^2,)\ x) = (G^1(q^1, z), G^2(q^2, y)) =$$

$$= (G^1(q^1, \varphi(x, v)), G^2(q^2, y)) =$$

$$= (G^1(q^1, \varphi(x, F^2(q^2))), G^2(q^2, F^1(q^1, \varphi(x, F^2(q^2))))) = \tilde{s},$$

где $\tilde{s}$ – состояние, в которое переходит результирующий автомат из состояния s под действием входного символа x. Состояние $\tilde{s}$ определяется как упорядоченная пара $\tilde{s} = (\tilde{q}^1, \tilde{q}^2)$, каждый из элементов которой есть результат вычисления соответствующей функции переходов компонентных автоматов.

7.6. Структурный автомат

7.6.1. Полнота структурных автоматов

Как упоминалось в разделе 7.1.1, все автоматы делятся на абстрактные и структурные. Введение в рассмотрение структурных автоматов связано не только со структурностью алфавитов. Если представить, что автомат – это «черный ящик», реализующий, «вычисляющий» надлежащим образом функции переходов и выхода, то структурный автомат позволяет «заглянуть» внутрь этого черного ящика с целью исследования его содержимого и изучения механизмов «вычисления». Этот подход во многом совпадает с подходом к исследованию логических схем, являющихся подходящей «вычислительной средой» для реализации логических функций в заданном базисе. Свойство полноты логического базиса гарантировало, что любая логическая функция может быть описана в его терминах и построена соответствующая логическая схема.

Подобный подход применяется и для исследования структурных автоматов. При этом полагается, что все каналы (входные, выходные и внутренние) могут находиться в одном из двух состояний: 0 или 1, а входные, выходные и внутренние символы представляются двоичными векторами длины n, m и k соответственно. Иногда такие структурные автоматы называют цифровыми автоматами.

Ответ на вопрос о полноте для структурных (цифровых) автоматов дает следующая теорема.

ТЕОРЕМА. Совокупность структурных элементов *структурно полна*, если она содержит:

– по крайней мере, один автомат Мура с нетривиальной памятью, обладающий полной системой переходов и полной системой выходов;

– функционально полную систему логических элементов.

Нетривиальные автоматы Мура ($|Q|>1$), обладающие полной системой переходов и полной системой выходов, называют ***автоматами памяти***.

Функционально полная система переходов автомата памяти означает, что функция переходов $G(q, x)$ обладает следующим свойством: для любых двух состояний q_i и q_j найдется такой входной символ x, для которого $G(q_i, x)=q_j$. Другими словами, в автоматах с полной системой переходов возможны переходы из любого состояния в любое.

Полная система выходов для автомата памяти означает, что существует взаимно однозначное соответствие между символами выходного алфавита Y и символами внутреннего алфавита Q. В результате значение выходного символа однозначно определяет состояние автомата памяти.

Ниже приведены примеры автоматов памяти, которые имеют минимальную сложность.

Автомат задержки

Входной алфавит $X=\{0, 1\}$.

Выходной алфавит $Y=\{0, 1\}$.

Внутренний алфавит $Q=\{q_0, q_1\}$.

Функция переходов и функция выхода определяются по его графическому представлению (рис. 7.12).

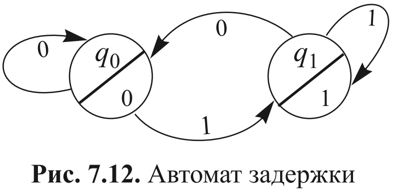


Рис. 7.12. Автомат задержки

Свое название автомат получил за то, что его состояние и, следовательно, значение выходного символа повторяют значение входного символа.

Автомат памяти является математической моделью структурного элемента – D-триггера (рис. 7.13).

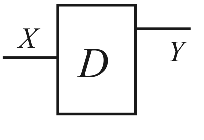


Рис. 7.13. Структурный элемент – D-триггер

Для целей анализа и синтеза на вход этого элемента следует подавать входной символ 1 всякий раз, когда необходимо перейти в состояние q_1, либо подавать символ 0, если необходимо перейти в состояние q_0. Эти условия определяют функцию возбуждения ψ: $X \times Y \rightarrow Y$ автомата задержки, которая явно не зависит от состояния (табл. 7.15).

Таблица 7.15

Функция возбуждения автомата задержки

X \ Y	0	1
0	0	0
1	1	1

Автомат со счетным входом

Входной алфавит $X = \{0, 1\}$.

Выходной алфавит $Y = \{0, 1\}$.

Внутренний алфавит $Q = \{q_0, q_1\}$.

Функция переходов и функция выхода приведена на рис. 7.14, *а*.

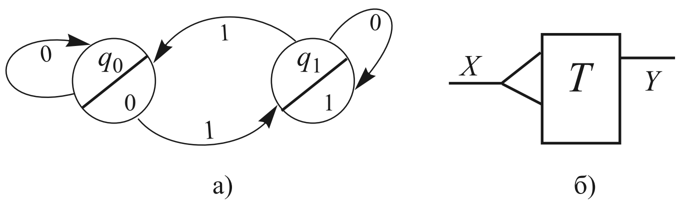


Рис. 7.14. Автомат со счетным входом

Свое название автомат получил за то, что состояние и, следовательно, значение выходного символа изменяется на противоположное при единичном значении входного символа. Входной символ 0 не изменяет состояния этого автомата.

Автомат со счетным входом является математической моделью структурного элемента, называемого T-триггером (рис. 7.14, *б*).

Для целей анализа и синтеза на вход этого элемента следует подавать входной символ 1 всякий раз, когда необходимо изменить текущее состояние. Входной символ 0 не изменяет текущее состояние. Эти условия определяют функцию возбуждения ψ: $X \times Y \to Y$ автомата (табл. 7.16).

Таблица 7.16

Функция возбуждения автомата со счетным входом

$X \backslash Y$	0	1
0	0	1
1	1	0

Можно привести еще много примеров автоматов памяти, которые имеют практическую реализацию. Каждый из таких автоматов в совокупности с функционально полной системой логических элементов позволяет строить структурные автоматы, подобно тому как ранее строились логические схемы.

7.6.2. Задача анализа и синтеза структурного автомата

Все структурные автоматы можно разделить на два класса:

– автоматы с распределенной памятью;

– автоматы с сосредоточенной памятью.

Здесь рассматриваются автоматы только с сосредоточенной памятью. На рис. 7.15 приведена схема такого автомата.

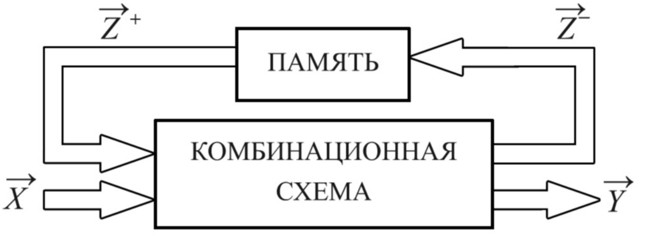


Рис. 7.15. Автомат с сосредоточенной памятью

Такой автомат условно состоит из двух частей: памяти и комбинационной схемы. Комбинационная схема не обладает памятью. Двоичный вектор $\overrightarrow{X} = (x_1, x_2, \ldots, x_n)$ задает состояние n-входных каналов. Двоичный вектор $\overrightarrow{Y} = (y_1, y_2, \ldots, y_m)$ задает состояние m-выходных каналов. Двоичный вектор $\overrightarrow{Z}^{+} = (z_1^{+}, z_2^{+}, \ldots, z_k^{+})$ задает состояние k-каналов памяти структурного автомата. Двоичный вектор $\overrightarrow{Z}^{-} = (z_1^{-}, z_2^{-}, \ldots, z_k^{-})$ задает состояние k-каналов возбуждения элементов памяти.

В качестве компонентного элемента в соответствии с теоремой о структурной полноте могут рассматриваться автоматы Мура с полной системой переходов и полной системой выходов, примеры которых были рассмотрены в разделе 7.6.2.

Комбинационная схема предназначена для «вычисления» векторных логических функций выхода и функций возбуждения элементов памяти в зависимости от состояния входного вектора $\overrightarrow{X}$ и вектора состояний памяти $\overrightarrow{Z}^{+}$:

$\overrightarrow{Y} = \overrightarrow{\varphi}(\overrightarrow{Z}^{+}, \overrightarrow{X})$ – функции выхода комбинационной схемы;

$\overrightarrow{Z}^{-} = \overrightarrow{\psi}(\overrightarrow{Z}^{+}, \overrightarrow{X})$ – функции возбуждения элементов памяти.

Комбинационная схема на логическом уровне представляется либо системой логических схем, либо программируемой памятью с соответствующей прошивкой.

Задача анализа структурного автомата

Дано: структурный автомат (рис. 7.16).

Для определения структурного автомата необходимо задать:

– тип элемента памяти (предполагается, что все они однотипны);
– функции возбуждения элементов памяти ($\overrightarrow{\psi}$);
– функции выхода ($\overrightarrow{\varphi}$).

Функции $\overrightarrow{\varphi}$ и $\overrightarrow{\psi}$ определяют комбинационную схему.

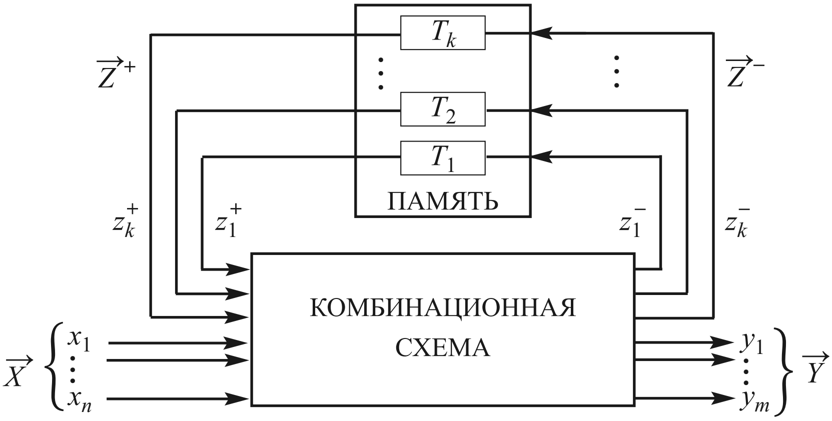


Рис. 7.16. Структурный автомат с заданным типом памяти

Определить (найти): множество состояний Q заданного автомата и функции выхода и переходов, которые реализуются заданным автоматом:

– G: $Q\times X\to Q$;

– F: $Q\times X\to Y$.

<u>Решение</u>. Для решения задачи анализа необходимо:

1. Найти множество состояний Q структурного автомата.
2. Найти функцию переходов G структурного автомата.
3. Найти функцию выхода F структурного автомата.

Для нахождения множества состояний структурного автомата необходимо по аргументной части задания функции $\vec{\varphi}$ выделить все возможные двоичные векторы $\vec{Z}^+$. Множество таких векторов определяет равномощное множество состояний $Q=\{q_1, q_2, \ldots\}$ анализируемого автомата. Каждому значению вектора $\vec{Z}^+$ ставится в соответствие символ q из Q.

Для нахождения функции переходов структурного автомата необходимо использовать функцию $\vec{\psi}$ и свойства элемента памяти (функцию возбуждения элемента памяти). С учетом того, что функции возбуждения элементов памяти определяются выходами

z_1^-, z_2^-, …, z_k^- комбинационной схемы, для каждого i-го элемента памяти определяется новое состояние в зависимости от его типа и исходного состояния.

Если в качестве элемента памяти используется D-триггер, при $z_i^-=0$, i-й элемент памяти переходит в состояние 0, а при $z_i^-=1$, i-й элемент памяти переходит в состояние 1.

Если в качестве элемента памяти используется T-триггер, то при $z_i^-=0$, i-й элемент памяти не изменяет свое состояние, а при $z_i^-=1$, i-й элемент памяти изменяет свое состояние на противоположное.

Пример.

Задан структурный автомат (рис. 7.17).

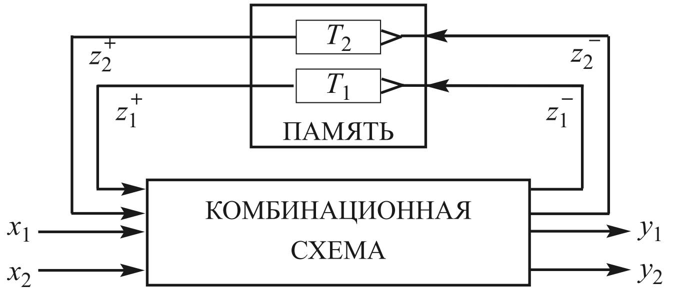


Рис. 7.17. Пример структурного автомата

Автомат имеет два входных канала x_1 и x_2, два выходных канала y_1 и y_2 и два элемента памяти типа триггера со счетным входом. Комбинационная схема реализует функции $\vec{\psi}$ и $\vec{\varphi}$, представленные в табл. 7.17.

Определить: множество состояний Q заданного автомата и функции выхода и переходов, которые реализуются заданным структурным автоматом:

– G: $Q\times X\rightarrow Q$;

– F: $Q\times X\rightarrow Y$.

Найденные функции представить в табличном виде.

Таблица 7.17

Функции $\vec{\psi}$ и $\vec{\varphi}$

z_1^+	z_2^+	x_1	x_2	$\vec{\psi}$		$\vec{\varphi}$	
				z_1^-	z_2^-	y_1	y_2
0	0	0	0	0	1	0	1
0	0	0	1	0	0	1	0
0	0	1	0	1	0	0	0
0	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1
0	1	0	1	0	1	1	0
0	1	1	0	1	0	0	1
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	0	1	0	1	1	0
1	0	1	0	1	0	0	1
1	0	1	1	1	1	1	1
1	1	0	0	1	1	1	1
1	1	0	1	0	1	0	0
1	1	1	0	0	0	0	1
1	1	1	1	1	0	1	0

Решение. Каналы z_1^+ и z_2^+ определяют все возможные состояния памяти в структурном алфавите (z_1^+, z_2^+). Таких состояний, как следует из первых двух столбцов табл. 7.17, четыре: 00, 01, 10 и 11. Введем абстрактный алфавит Q состояний автомата $Q=\{q_1, q_2, q_3, q_4\}$, где q_1 соответствует 00, q_2 соответствует 01, q_3 соответствует 10, q_4 соответствует 11.

В табл. 7.18 добавлены два столбца: $Q(t-1)$ и $Q(t)$. Первый из этих столбцов является перекодированием столбцов z_1^+ и z_2^+, определяющих состояние автомата в абстрактном алфавите $Q=\{q_1, q_2, q_3, q_4\}$. Столбец $Q(t)$ определяется из свойств функции возбуждения элемента памяти со счетным входом: состояние элемента памяти изменяется

Таблица 7.18

Функции $\vec{\psi}$ и $\vec{\varphi}$

z_1^+	z_2^+	x_1	x_2	$\vec{\psi}$		$\vec{\varphi}$		$Q(t-1)$	$Q(t)$
				z_1^-	z_2^-	y_1	y_2		
0	0	0	0	0	1	0	1	q_1	q_2 (01)
0	0	0	1	0	0	1	0	q_1	q_1 (00)
0	0	1	0	1	0	0	0	q_1	q_3 (10)
0	0	1	1	1	1	1	1	q_1	q_4 (11)
0	1	0	0	1	1	1	1	q_2	q_3 (10)
0	1	0	1	0	1	1	0	q_2	q_1 (00)
0	1	1	0	1	0	0	1	q_2	q_4 (11)
0	1	1	1	1	1	1	1	q_2	q_3 (10)
1	0	0	0	1	1	1	1	q_3	q_2 (01)
1	0	0	1	0	1	1	0	q_3	q_4 (11)
1	0	1	0	1	0	0	1	q_3	q_1 (00)
1	0	1	1	1	1	1	1	q_3	q_2 (01)
1	1	0	0	1	1	1	1	q_4	q_1 (00)
1	1	0	1	0	1	0	0	q_4	q_3 (10)
1	1	1	0	0	0	0	1	q_4	q_4 (11)
1	1	1	1	1	0	1	0	q_4	q_2 (01)

всякий раз, когда на вход ему поступает 1. Это правило действует для каждого такого элемента.

При анализе первой строки таблицы отмечается, что $z_1^-=0$, а $z_2^-=1$. Это означает, что состояние первого элемента памяти на данном переходе не изменяется, а состояние второго элемента памяти должно измениться на противоположное. Новое состояние памяти в структурном и в абстрактном алфавите приведено в столбце $Q(t)$. Подобный анализ проводится для каждой строки табл. 7.18.

Для представления результатов анализа строится таблица для функции переходов (табл. 7.19, *а*) и для функции выхода (табл. 7.19, *б*).

Таблица 7.19

Функция переходов G и функция выхода F

$X \backslash Q$	q_1	q_2	q_3	q_4
00	q_2	q_3	q_2	q_1
01	q_1	q_1	q_4	q_3
10	q_3	q_4	q_1	q_4
11	q_4	q_3	q_2	q_2

а) G: $Q\times X\rightarrow Q$

$X \backslash Q$	q_1	q_2	q_3	q_4
00	01	11	11	11
01	10	10	10	00
10	00	01	01	01
11	11	11	11	10

б) F: $Q\times X\rightarrow Y$

Задача синтеза структурного автомата

Дано: структурный автомат, заданный:

- функциями переходов и выхода в структурном алфавите
  - G: $Q\times X\rightarrow Q$;
  - F: $Q\times X\rightarrow Y$;
- типом элемента памяти;
- функционально полным набором логических элементов (структурным логическим базисом).

Определить (найти): структуру автомата (множество элементов памяти заданного типа, множество структурных логических элементов и схему их соединения), которая обеспечивает требуемое поведение, определяемое заданными функциями G: $Q\times X\rightarrow Q$ и F: $Q\times X\rightarrow Y$.

Задача синтеза является обратной по отношению к задаче анализа и не имеет однозначного решения. Для сужения пространства возможных решений используют критерии, которые позволяют производить оценку качества решения. В качестве подобных критериев могут выступать такие показатели, как стоимость синтезируемого автомата, его надежность, трудоемкость нахождения решения и т.п. В рамках данного учебного пособия эта задача не может быть детально рассмотрена. Здесь предлагается рассмотреть инженерный подход к задаче синтеза, который основан на модели структурного автомата (рис. 7.16).

При таком подходе для решения задачи синтеза автомата необходимо:

- определить требуемое число элементов памяти (задать память автомата);
- построить функции выхода ($\vec{\varphi}$) и возбуждения ($\vec{\psi}$);
- построить логические схемы в заданном базисе для функций $\vec{\varphi}$ и $\vec{\psi}$ (определить комбинационную схему автомата).

Замечание. Иногда состояния автомата в исходных данных для синтеза представлены в абстрактном, а не в структурном двоичном алфавите. В этом случае необходимо предварительно провести кодирование внутренних состояний двоичными векторами фиксированной длины. Длина кода должна быть достаточной для однозначного кодирования всех состояний автомата.

Пример.

Задан автомат (рис. 7.18), у которого входной и выходной алфавиты – структурные.

Рис. 7.18. Пример автомата

Функции переходов G и функция выходов F для этого автомата приведена в табл. 7.20, *а* и табл. 7.20, *б* соответственно.

Таблица 7.20

Функция переходов (а) и функция выхода (б) заданного автомата

$X \backslash Q$	q_1	q_2	q_3	q_4
0	q_3	q_3	q_2	q_1
1	q_2	q_4	q_1	q_3

а) G: $Q \times X \rightarrow Q$

$X \backslash Q$	q_1	q_2	q_3	q_4
0	11	10	11	00
1	01	10	00	01

б) F: $Q \times X \rightarrow Y$

В качестве элемента памяти использовать триггер со счетным входом, а в качестве логического элемента использовать элемент Шеффера на два входа.

Определить (найти): структуру (логическую схему) автомата в заданном структурном базисе, который обеспечивает требуемое поведение, определяемое заданными функциями F и G (табл. 7.20).

Решение.

Алфавит состояния заданного автомата не является структурным. Для приведения его к структурному виду необходимо провести кодирование состояний двоичными векторами. Минимальная длина такого вектора равна двум, и, следовательно, число элементов памяти, необходимых для синтеза структурного автомата, равно 2. Пусть проведено следующее кодирование состояний:

$q_1 \leftrightarrow 00$, $q_2 \leftrightarrow 01$, $q_3 \leftrightarrow 10$, $q_4 \leftrightarrow 11$.

На рис. 7.19 приведена обобщенная схема искомого автомата.

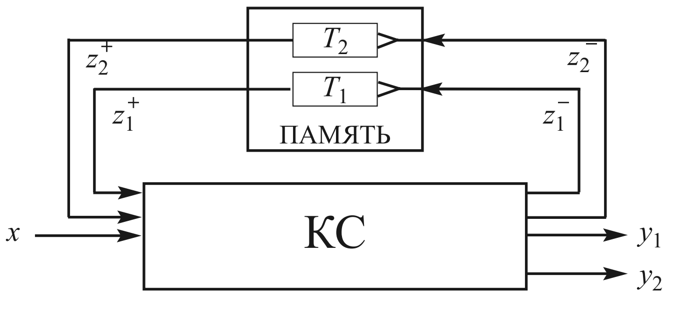


Рис. 7.19. Обобщенная структура автомата

Для решения задачи синтеза необходимо:

- определить функции выхода КС: $\vec{Y} = \vec{\varphi}(\vec{Z}^+, \vec{X})$;
- определить функции возбуждения элементов памяти $\vec{Z}^- = \vec{\psi}(\vec{Z}^+, \vec{X})$;
- построить логические схемы в заданном базисе Шеффера, «вычисляющие» функции $\vec{\varphi}$ и $\vec{\psi}$.

В табл. 7.21 приведены функции F и G с учетом проведенного кодирования состояний.

Таблица 7.21

Функции F и G

$Q(t-1)$		X	$\vec{\varphi}$		$Q(t)$		$\vec{\psi}$	
z_1^+	z_2^+		y_1	y_2	z_1^+	z_2^+	z_1^-	z_2^-
0	0	0	1	1	1	0	1	0
0	0	1	0	1	0	1	0	1
0	1	0	1	0	1	0	1	1
0	1	1	1	0	1	1	1	0
1	0	0	1	1	0	1	1	1
1	0	1	0	0	0	0	1	0
1	1	0	0	0	0	0	1	1
1	1	1	0	1	1	0	0	1

В этой таблице столбец y_1 определяет функцию $y_1 = \varphi_1(z_1^+, z_2^+, x)$, $y_2 = \varphi_2(z_1^+, z_2^+, x)$.

Функции возбуждения элементов памяти $z_1^- = \psi_1(z_1^+, z_2^+, x)$ и $z_1^- = \psi_2(z_1^+, z_2^+, x)$ находятся из условия использования элементов памяти типа триггера со счетным входом. На вход такого триггера следует подавать 1, если необходимо изменить его состояние.

В первой строке табл. 7.21, соответствующей переходу из состояния 00 в состояние 10, происходит изменение состояния первого элемента памяти. Для этой строки $z_1^- = 1$, $z_2^- = 0$.

Во второй строке $z_1^- = 0$, $z_2^- = 1$.

В третьей строке $z_1^- = 1$, $z_2^- = 1$.

В четвертой строке $z_1^- = 1$, $z_2^- = 0$.

В пятой строке $z_1^- = 1$, $z_2^- = 1$.

В шестой строке $z_1^- = 1$, $z_2^- = 0$.

В седьмой строке $z_1^- = 1$, $z_2^- = 1$.

В восьмой строке $z_1^- = 0$, $z_2^- = 1$.

Эти два столбца и определяют функции $\psi_1(z_1^+, z_2^+, x)$ и $\psi_2(z_1^+, z_2^+, x)$.

Заключительный этап синтеза предполагает построение логических схем с использованием элемента И-НЕ функций $\varphi_1(z_1^+, z_2^+, x)$, $\varphi_2(z_1^+, z_2^+, x)$, $\psi_1(z_1^+, z_2^+, x)$ и $\psi_2(z_1^+, z_2^+, x)$, которые определят структуру комбинационной схемы автомата.

• Задачи и упражнения

① Для заданного автомата Мили $A=<X, Y, Q, F, G>$, где $X=\{x_1, x_2, x_3\}$, $Y=\{y_1, y_2, y_3\}$, $Q=\{q_1, q_2, q_3\}$, а функции F и G определены в следующей таблице:

F: $Q\times X\to Y$				G: $Q\times X\to Q$		
$X \backslash Q$	q_1	q_2	q_3	q_1	q_2	q_3
x_1	y_1	y_3	y_2	q_2	q_3	q_3
x_2	y_1	y_1	y_2	q_1	q_1	q_2
x_3	y_3	y_3	y_3	q_3	q_1	q_1

q Определить последовательность выходных символов, которая является реакцией автомата на последовательность входных символов $\xi_x=x_1x_1x_1x_3x_2x_3x_3$, если его начальное состояние $q(0)=q_1$.

q Определить возможность получения на выходе автомата последовательности выходных символов $\xi_y=y_3y_2y_2y_2y_1y_3y_3y_1$, указав допустимое начальное состояние и соответствующую последовательность входных символов.

q Построить эквивалентный автомат Мура.

② Построить автомат Мили, который преобразует произвольную двоичную входную последовательность в двоичную выходную последовательность. I-й символ выходной последовательности ξ_y равен 1, если входная последовательность $\xi_x=x(1), x(2), \ldots, x(i)$ заканчивается символами 101. В противном случае значение *i*-го выходного символа равно 0.

③ Для заданного автомата Мили

A=<X, Y, Q, F, G>, где $X=\{x_1, x_2\}$, $Y=\{y_1, y_2\}$, $Q=\{q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9\}$, а функции F и G заданы в табличном виде:

$F: Q \times X \to Y$									
X \ Q	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9
0	y_2	y_1	y_1	y_1	y_1	y_2	y_1	y_2	y_2
1	y_1	y_2	y_2	y_2	y_2	y_1	y_2	y_1	y_1

$G: Q \times X \to Q$									
X \ Q	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9
0	q_4	q_7	q_5	q_6	q_1	q_5	q_8	q_7	q_4
1	q_6	q_6	q_1	q_3	q_3	q_8	q_2	q_1	q_2

Определить (найти):

q классы эквивалентных состояний;

q минимальный автомат Мили, эквивалентный исходному автомату.

④ На любом языке программирования определить структуру данных и написать процедуру (процедуры), которая программно моделирует поведение:

q произвольного автомата Мили;

q произвольного автомата Мура.

8. ЭЛЕМЕНТЫ НЕЧЕТКОЙ ЛОГИКИ И НЕЧЕТКИХ МНОЖЕСТВ

Традиционную бинарную логику, отдавая дань уважения важности, стройности и четкости, часто критикуют за ее определенную ограниченность, связанную с возможными значениями логических объектов: истина (1) и ложь (0). Хотя такое деление на истину и ложь, черное и белое имеет большое теоретическое и практическое значение, исследователям хотелось бы иметь более «тонкий» логический инструмент, позволяющий работать с «оттенками», а не только с «черным и белым». Однако обнаружилась несостоятельность бинарной логики при оценке логической значимости множества утверждений, с которыми довольно легко справляется человечество в своей практике. Так, каждый из нас примерно представляет себе, чем отличается утверждение «самолет летит низко» от утверждения «самолет летит высоко» или от утверждения «самолет летит на средней высоте». Можно привести множество подобных утверждений. А как следует трактовать в бинарной логике указание инструктора пилоту учебного самолета «возьмите левее»?

Ответ на эти и другие подобные вопросы позволяет получить нечеткая логика, которая была предложена Л. Заде в 70-е годы. В качестве синонимов нечеткости используются термины «размытая, расплывчатая логика», «Fuzzy Logic» и др. Аппарат нечеткой логики находит применение при построении систем управления, моделирующих поведенческие аспекты человека, интеллектуальные системы управления. Известны реализации систем управления с нечеткой логикой. Примером может служить система управления автоматической трансмиссией в современных автомобилях, в которых в качестве управляющего воздействия используются нечеткие высказывания «экономичный режим вождения», «спортивный режим вождения», «режим вождения по скользкой дороге».

Основной объект нечеткой логики – нечеткое логическое высказывание. В качестве интуитивного определения этого объекта можно привести следующее.

> ***Нечеткое логическое высказывание*** – это повествовательное утверждение, мера истинности которого задается некоторой количественной величиной μ, для которой справедливо условие $0 \le \mu \le 1$. Эту величину называют значением, мерой истинности нечеткого высказывания.

В следующих примерах приведены нечеткие высказывания с указанием субъективного значения истинности этих утверждений.

- μ («для вывода автомобиля из заноса снижаются обороты двигателя до минимума»)$=0.5$.
- μ («высота 1000 метров над землей недостаточна для длительного полета дальнемагистральных самолетов»)$=0.9$.
- μ («150 км/час – нормальная скорость самолета при посадке в момент касания посадочной полосы аэродрома»)$=0.1$.

Значения истинности перечисленных выше высказываний носят субъективный характер и не имеют статистического или вероятностного обоснования.

Пусть a – некоторое нечеткое логическое высказывание, а μ_a – его мера истинности. Далее в тексте слово «логическое» в контексте «логическое высказывание» опускается, если это не вызывает двусмысленности.

Если $\mu_a=1$, то нечеткое высказывание a истинно.

Если $0.5 \le \mu_a \le 1$, то нечеткое высказывание a более истинно, чем ложно.

Если $0 \leq \mu_a \leq 0.5$, то нечеткое высказывание a более ложно, чем истинно.

Если $\mu_a = 0$, то нечеткое высказывание a ложно.

Если $\mu_a = 0.5$, то нечеткое высказывание a индифферентно относительно истинности и ложности.

Введенное понятие нечеткого логического высказывания «покрывает» рассмотренное ранее определение логического высказывания в бинарной логике. Действительно, если предположить, что для любого нечеткого высказывания a значение меры истинности μ_a принимает одно из двух возможных значений – ноль либо единица, то такое нечеткое высказывание будет трактоваться как четкое или просто как логическое высказывание.

По аналогии с бинарной логикой нечеткие логические высказывания бывают простыми (неразделимыми на части) и сложными (составными). Сложные нечеткие высказывания получаются из простых с помощью операций нечеткой логики, подобных операциям бинарной логики:

- операция логического сложения «$\vee$»;
- операция логического умножения «$\wedge$»;
- операция логического отрицания «$\overline{\ }$»;
- операция логического следования «$\rightarrow$»;
- логическая эквивалентность «$\leftrightarrow$»;
- сложение Жегалкина «$\oplus$».

Содержательная интерпретация этих операций в естественном языке полностью совпадает с интерпретацией в естественном языке соответствующих операций бинарной логики. «Арифметический» же аспект вычисления меры истинности логических операций для нечетких высказываний носит иной характер.

Пусть a и b – некоторые нечеткие логические высказывания, а μ_a и μ_b – мера истинности высказываний a и b соответственно.

Логическим сложением (дизъюнкцией) нечетких высказываний a и b является нечеткое высказывание $a \vee b$, значение истинности которого $\mu_{a \vee b}$ определяется как $\mu_{a \vee b} = \max\{\mu_a, \mu_b\}$.

Логическим умножением (конъюнкцией) нечетких высказываний a и b является нечеткое высказывание $a \wedge b$, значение истинности которого $\mu_{a \wedge b}$ определяется как $\mu_{a \wedge b} = \min\{\mu_a, \mu_b\}$.

Логическим отрицанием (инверсией) нечеткого высказывания a является нечеткое высказывание $\overline{a}$, значение истинности которого $\mu_{\overline{a}}$ определяется как $\mu_{\overline{a}} = 1 - \mu_a$.

Логическим следованием (импликацией) нечеткого высказывания b из нечеткого высказывания a является нечеткое высказывание $a \rightarrow b$, значение истинности которого $\mu_{a \rightarrow b}$ определяется как $\mu_{a \rightarrow b} = \max\{1 - \mu_a, \mu_b\}$. Схема вычислений становится понятной, если вспомнить, что в бинарной логике $a \rightarrow b$ равносильно $\overline{a} \vee b$.

Логической эквивалентностью нечетких высказываний a и b является нечеткое высказывание $a \leftrightarrow b$, значение истинности которого $\mu_{a \leftrightarrow b}$ определяется как $\mu_{a \leftrightarrow b} = \max\{\min\{1 - \mu_a, 1 - \mu_b\}, \min\{\mu_a, \mu_b\}\}$. Схема вычислений становится понятной, если вспомнить, что в бинарной логике $a \leftrightarrow b$ равносильно $\overline{a} \wedge \overline{b} \vee a \wedge b$.

Жегалкинским сложением нечетких высказываний a и b является нечеткое высказывание $a \oplus b$, значение истинности которого $\mu_{a \oplus b}$ определяется как $\mu_{a \oplus b} = \max\{\min\{1 - \mu_a, \mu_b\}, \min\{\mu_a, 1 - \mu_b\}\}$. Схема вычислений становится понятной, если вспомнить, что в бинарной логике $a \oplus b$ равносильно $\overline{a} \wedge b \vee a \wedge \overline{b}$.

В табл. 8.1 приведен пример вычисления меры истинности введенных логических операций при различных значениях истинности исходных нечетких высказываний a и b.

Вычисление меры истинности более сложных нечетких высказываний требует учитывать старшинство (приоритет) логических опе-

Таблица 8.1

Пример вычисления меры истинности для логических операций

μ_a	μ_b	$\mu_{a\vee b}$	$\mu_{a\wedge b}$	$\mu_{\bar{a}}$	$\mu_{\bar{b}}$	$\mu_{a\to b}$	$\mu_{a\leftrightarrow b}$	$\mu_{a\oplus b}$
0	0	0	0	1	1	1	1	0
0	0.2	0.2	0	1	0.8	1	0.8	0.2
0.2	0.3	0.3	0.2	0.8	0.7	0.8	0.7	0.3
0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
0.7	0.5	0.7	0.5	0.3	0.5	0.5	0.5	0.5
0.7	0.7	0.7	0.7	0.3	0.3	0.7	0.7	0.3
1	0.7	1	0.7	0	0.3	0.7	0.7	0.3
1	1	1	1	0	0	1	1	0

раций. Старшей считается операция, стоящая в скобках. Отрицание по старшинству приравнивается к скобкам. Затем с меньшим приоритетом выполняется операция логического умножения, затем – операция логического сложения. Все остальные операции имеют низший приоритет, а старшинство между ними регулируется скобками.

Пусть задано три нечетких логических высказывания a, b и c, для которых $\mu_a = 0.9$, $\mu_b = 0.3$ и $\mu_c = 0.8$. Требуется вычислить значение истинности нечеткого высказывания $\overline{a\vee\overline{b\to c}\wedge\overline{\bar{a}\leftrightarrow(\bar{b}\oplus c)}}$.

В соответствии со старшинством операций

$$\mu(\overline{a\vee\overline{b\to c}\wedge\overline{\bar{a}\leftrightarrow(\bar{b}\oplus c)}}) = 1-\mu(a\vee\overline{b\to c}\wedge\overline{\bar{a}\leftrightarrow(\bar{b}\oplus c)}) = 1-\max\{\mu(a),$$
$$\mu(\overline{b\to c}\wedge\overline{\bar{a}\leftrightarrow(\bar{b}\oplus c)})\}=1-\max\{\mu(a),\min\{\mu(\overline{b\to c}),\ \mu(\overline{\bar{a}\leftrightarrow(\bar{b}\oplus c)})\}\}=$$
$$=1-\max\{\mu(a),\ \min\{1-\mu(b\to c),\ 1-\mu(\bar{a}\leftrightarrow(\bar{b}\oplus c))\}\}=1-\max\{\mu(a),$$
$$\min\{1-\max\{1-\mu(b),\ \mu(c)\},\ 1-\max\{\min\{\mu(\bar{a}),\ \mu(\bar{b}\oplus c)\},\ \min\{\mu(a),$$
$$\mu(\overline{\bar{b}\oplus c})\}\}\}\}.$$

Значение истинности выражения $\mu(\overline{\bar{b}\oplus c})=1-\mu(\bar{b}\oplus c)$. После раскрытия и подстановки конкретных значений мера истинности выражения $\mu(\bar{b}\oplus c)$ равна $\max\{\min\{\mu(\bar{\bar{b}}),\ \mu(c)\},\ \min\{\mu(\bar{b}),\ \mu(\bar{c})\}\}=$ $=\max\{\min\{\mu(b),\ \mu(c)\},\ \min\{1-\mu(b),\ 1-\mu(c)\}\}=\max\{\min\{0.3,\ 0.8\},$ $\min\{1-0.3,\ 1-0.8\}\}=\max\{0.3,\ 0.2\}=0.3$, а $\mu(\overline{\bar{b}\oplus c})=0.7$.

Окончательно, $\mu(\overline{a\vee\overline{b\to c}\wedge\overline{\bar{a}\leftrightarrow(\bar{b}\oplus c)}})=$

$=1-\max\{\mu(a),\ \min\{1-\max\{1-\mu(b),\ \mu(c)\},\ 1-\max\{\min\{\mu(\bar{a}),$ $\mu(\bar{b}\oplus c)\},\min\{\mu(\bar{\bar{a}}),\ \mu(\overline{\bar{b}\oplus c})\}\}\}\}=1-\max\{0.9,\min\{1-\max\{1-0.3,0.8\},$ $1-\max\{\min\{0.1,\ 0.3\},\ \min\{0.9,\ 0.7\}\}\}\}=1-\max\{0.9,\ 0.2\}\}=1-$ $-0.9=0.1$.

Нечеткое логическое высказывание, в котором не задана мера истинности для образующих его простых высказываний, по аналогии с бинарной логикой называют нечеткой логической высказывательной функцией. Роль переменных в ней выполняют простые нечеткие логические высказывания. Возможное значение такой переменной и самой функции ограничено интервалом от 0 до 1. Если в качестве меры истинности используется фиксированный конечный набор возможных значений истинности, шкала меры истинности нечеткой высказывательной функции дискретна. В противном случае шкала непрерывна.

В общем случае нечеткая логическая функция от n нечетких переменных определяется как $f(x_1, x_2,\ldots, x_n)$: $[0,1]^n\to[0,1]$. В отличие от бинарной логики, в которой каждая переменная и сама функция могут принимать только два значения, в нечеткой логике переменные и функция могут принимать любое значение из интервала истинности. Нечеткая функция превращается в нечеткое высказывание в результате интерпретации, при которой каждой нечеткой переменной сопоставляется некоторое логическое высказывание с указанием значения истинности.

Важным для логики высказываний, в том числе и для нечеткой логики, является понятие равносильности логических функций и связанные с этим понятием правила эквивалентных преобразований. В частности, выполняются ли для нечеткой логики равносильности, справедливые для бинарной логики такие, как идемпотентность, коммутативность, ассоциативность, дистрибутивность, правило де Моргана, свойства 0 и 1?

Пусть $f(x_1, x_2,\ldots, x_n)$ и $g(x_1, x_2,\ldots, x_n)$ – произвольные нечеткие функции n нечетких переменных $x_1, x_2,\ldots, x_n$, а $\mu_f(\mu_1, \mu_2,\ldots, \mu_n)$ и $\mu_g(\mu_1, \mu_2,\ldots, \mu_n)$ – мера истинности функций f и g в зависимости от распределения значений истинности переменных $x_1, x_2,\ldots, x_n$ соответственно. Две функции $f(x_1, x_2,\ldots, x_n)$ и $g(x_1, x_2,\ldots, x_n)$ равносильны, если для каждого распределения допустимых значений истинности $\mu_1, \mu_2,\ldots, \mu_n$ переменных $x_1, x_2,\ldots, x_n$ справедливо $\mu_f(\mu_1, \mu_2,\ldots, \mu_n)= =\mu_g(\mu_1, \mu_2,\ldots, \mu_n)$.

<u>Замечание</u>. С учетом того, что равносильность нечетких функций является также нечетким высказыванием $f(x_1, x_2,\ldots, x_n)\leftrightarrow g(x_1, x_2,\ldots, x_n)$, для равносильности достаточно, чтобы для каждого допустимого набора значений истинности $\mu_1, \mu_2,\ldots, \mu_n$ переменных $x_1, x_2,\ldots, x_n$ выполнялось условие $\mu(f(x_1, x_2,\ldots, x_n)\leftrightarrow g(x_1, x_2,\ldots, x_n))\geq 0{,}5$.

<u>ТЕОРЕМА</u>. Пусть $f(x_1, x_2,\ldots, x_n)$, $g(x_1, x_2,\ldots, x_n)$ и $h(x_1, x_2,\ldots, x_n)$ – произвольные нечеткие логические функции. Тогда справедливы следующие равносильности:

1. Идемпотентность:

$f(x_1, x_2,\ldots, x_n)\vee f(x_1, x_2,\ldots, x_n)=f(x_1, x_2,\ldots, x_n)$;

$f(x_1, x_2,\ldots, x_n)\wedge f(x_1, x_2,\ldots, x_n)=f(x_1, x_2,\ldots, x_n)$.

2. Коммутативность:

$f(x_1, x_2,\ldots, x_n)\vee g(x_1, x_2,\ldots, x_n)=g(x_1, x_2,\ldots, x_n)\vee f(x_1, x_2,\ldots, x_n)$;

$f(x_1, x_2,\ldots, x_n)\wedge g(x_1, x_2,\ldots, x_n)=g(x_1, x_2,\ldots, x_n)\wedge f(x_1, x_2,\ldots, x_n)$.

3. Ассоциативность:

$(f(x_1, x_2,\ldots, x_n)\vee g(x_1, x_2,\ldots, x_n))\vee h(x_1, x_2,\ldots, x_n)=$
$=f(x_1, x_2,\ldots, x_n)\vee(g(x_1, x_2,\ldots, x_n)\vee h(x_1, x_2,\ldots, x_n))$;

$(f(x_1, x_2,\ldots, x_n)\wedge g(x_1, x_2,\ldots, x_n))\wedge h(x_1, x_2,\ldots, x_n)=$
$=f(x_1, x_2,\ldots, x_n)\wedge(g(x_1, x_2,\ldots, x_n)\wedge h(x_1, x_2,\ldots, x_n))$.

4. Дистрибутивность:

$$f(x_1, x_2,\ldots, x_n)\wedge(g(x_1, x_2,\ldots, x_n)\vee h(x_1, x_2,\ldots, x_n))=$$
$$=f(x_1, x_2,\ldots, x_n)\wedge g(x_1, x_2,\ldots, x_n)\vee f(x_1, x_2,\ldots, x_n)\wedge h(x_1, x_2,\ldots, x_n);$$
$$f(x_1, x_2,\ldots, x_n)\vee g(x_1, x_2,\ldots, x_n)\wedge h(x_1, x_2,\ldots, x_n)=$$
$$=(f(x_1, x_2,\ldots, x_n)\vee(g(x_1, x_2,\ldots, x_n))\wedge(f(x_1, x_2,\ldots, x_n)\vee h(x_1, x_2,\ldots, x_n)).$$

5. Де Моргана:

$$\overline{f(x_1, x_2,\ldots, x_n)\vee g(x_1, x_2,\ldots, x_n)}=\overline{f(x_1, x_2,\ldots, x_n)}\wedge\overline{g(x_1, x_2,\ldots, x_n)};$$
$$\overline{f(x_1, x_2,\ldots, x_n)\wedge g(x_1, x_2,\ldots, x_n)}=\overline{f(x_1, x_2,\ldots, x_n)}\vee\overline{g(x_1, x_2,\ldots, x_n)}.$$

6. Двойное отрицание:

$$\overline{\overline{f(x_1, x_2,\ldots, x_n)}}=f(x_1, x_2,\ldots, x_n).$$

В качестве примера ниже приведено доказательство равносильности первого свойства дистрибутивности:

$$f(x_1, x_2,\ldots, x_n)\wedge(g(x_1, x_2,\ldots, x_n)\vee h(x_1, x_2,\ldots, x_n))=$$
$$=f(x_1, x_2,\ldots, x_n)\wedge g(x_1, x_2,\ldots, x_n)\vee f(x_1, x_2,\ldots, x_n)\wedge h(x_1, x_2,\ldots, x_n).$$

Для доказательства необходимо показать, что при произвольных значениях истинности μ_f, μ_g, и μ_h функций f, g и h выполняется равенство $\min\{\mu_f, \max\{\mu_g, \mu_h\}\}=\max\{\min\{\mu_f, \mu_g\},\min\{\mu_f, \mu_h\}\}$.

Пусть $\max\{\mu_g, \mu_h\}=\mu_g$. Тогда $\min\{\mu_f, \max\{\mu_g, \mu_h\}\}=\min\{\mu_f, \mu_g\}$. Учитывая, что $\min\{\mu_f, \mu_g\}\geq\min\{\mu_f, \mu_g\}$ и, следовательно, $\max\{\min\{\mu_f, \mu_g\}, \min\{\mu_f, \mu_h\}\}=\min\{\mu_f, \mu_g\}$, показано, что левая и правая части равенства сводятся к одному и тому же значению. В силу симметричности вхождения μ_g, μ_h в $\max\{\mu_g, \mu_h\}$ схема доказательства остается неизменной в предположении, что $\max\{\mu_g, \mu_h\}=\mu_h$.

В нечеткой логике не выполняются некоторые известные в бинарной логике равносильности, такие, например, как $f(x_1, x_2,\ldots, x_n)\vee\overline{f(x_1, x_2,\ldots, x_n)}=1$ или $f(x_1, x_2,\ldots, x_n)\wedge\overline{f(x_1, x_2,\ldots, x_n)}=0$. Для доказательства этого факта можно привести пример, опровергающий это свойство.

Пусть $\mu_f=0.3$, тогда $\mu(f(x_1, x_2,\ldots, x_n)\vee\overline{f(x_1, x_2,\ldots, x_n)})=\max\{\mu_f, 1-\mu_f\}=\max\{0.3, 1-0.3\}=0.7\neq 1$. Что и требовалось доказать.

Введение в теорию и практику категории нечеткости повлияло не только на развитие математической логики, но и на развитие теории множеств. Наряду с понятием множества появилось понятие нечеткого множества. Основное различие между множеством и нечетким множеством заключено в характеристическом свойстве, которое определяет истинность логического высказывания «элемент m принадлежит множеству M». Сокращенно: «$m\in M$». Если это высказывание принимает одно из двух возможных значений – истина либо ложь (0 либо 1), то соответствующее множество четкое. Для нечетких множеств характеристическое свойство определяет меру, значение истинности высказывания «$m\in M$» является нечетким логическим высказыванием и может принимать произвольное значение в интервале от 0 до 1. Часто ее называют функцией принадлежности элемента множеству.

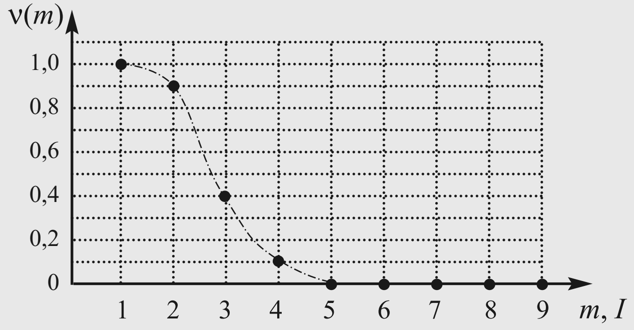
Пусть ν: $\mathrm{M}\to[0, 1]$ – функция принадлежности элемента m множеству M, которое определено на универсуме I. Тогда для каждого m ($m\in I$) величина $\nu(\mathrm{m})$ задает меру истинности высказывания «$m\in M$»: $M=\{m|0<\nu(m)\leq 1$ и $m\in I\}$. Таким образом, нечеткое множество – это множество пар вида $\langle m, \nu(m)\rangle$, где первый элемент – элемент универсума, а второй – мера истинности принадлежности этого элемента к множеству. Элементы универсума, для которых значение функции ν равно 0, не содержатся в множестве M.

В рассмотренных ниже примерах в качестве универсума используется конечное множество натуральных чисел от 1 до 9: $I=\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Множество $M_1=\{m|m\in I$ и $\nu_1(m)$: «число m не намного превосходит 1»$\}$. На рис. 8.1 приведены нечеткое множество M_1, в котором рядом с каждым элементом m указано значение истинности его принадлежности к M_1, графическое и табличное представление функции $\nu(m)$ принадлежности нечеткому множеству M_1.

а) $M_1 = \{(1, 1.0), (2, 0.9), (3, 0.4), (4, 0.1)\}$

б)



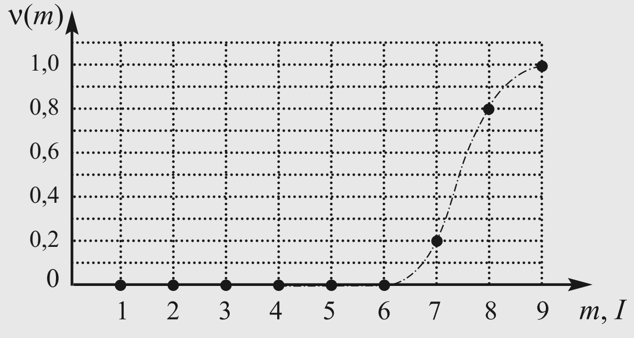
в)

m	1	2	3	4	5	6	7	8	9
$\nu(m)$	1,0	0,9	0,4	0,1	0	0	0	0	0

Рис. 8.1. Пример нечеткого множества: а) множество M_1;
б) графическое представление функции $\nu(m)$;
в) табличное представление функции $\nu(m)$

а) $M_2 = \{(7, 0.2), (8, 0.8), (9, 1.0)\}$

б)



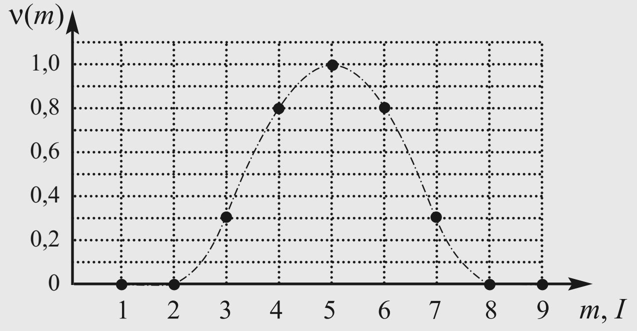
в)

m	1	2	3	4	5	6	7	8	9
$\nu(m)$	0	0	0	0	0	0	0,2	0,8	1,0

Рис. 8.2. Пример нечеткого множества: а) множество M_2;
б) графическое представление функции $\nu(m)$;
в) табличное представление функции $\nu(m)$

а) $M_1 = \{(3,\ 0.3),\ (4,\ 0.8),\ (5,\ 1.0),\ (6,\ 0.8),\ (7,\ 0.3)\}$

б)



в)

m	1	2	3	4	5	6	7	8	9
$\nu(m)$	0	0	0,3	0,8	1,0	0,8	0,3	0	0

Рис. 8.3. Пример нечеткого множества: а) множество M_3; б) графическое представление функции $\nu(m)$; в) табличное представление функции $\nu(m)$

Множество $M_2 = \{m \mid m \in I$ и $\nu_2(m)$: «число m значительно больше 1»$\}$. На рис. 8.2 приведены нечеткое множество M_2, в котором рядом с каждым элементом m указано значение истинности его принадлежности к M_2, графическое и табличное представление функции $\nu(m)$ принадлежности нечеткому множеству M_2.

Множество $M_3 = \{m \mid m \in I$ и $\nu_3(m)$: «число m близко к среднему значению универсума»$\}$. На рис. 8.3 приведены нечеткое множество M_3, в котором рядом с каждым элементом m указано значение истинности его принадлежности к M_3, графическое и табличное представление функции $\nu(m)$ принадлежности нечеткому множеству M_3.

Для нечетких множеств определено понятие включения. Пусть A и B – некоторые нечеткие множества, определенные на общем универсуме, для которых ν_A и ν_B – функции принадлежности для A и B соответственно.

Нечеткое множество A является нечетким подмножеством нечеткого множества B, если для каждого элемента m универсума $\nu_a(m) \leq \nu_b(m)$.

Для нечетких множеств определены операции объединения ($\cup$), пересечения ($\cap$) и дополнения ($\overline{\ }$) до универсума.

Объединение ($\cup$). Объединение двух нечетких множеств $\langle M_a, \nu_a \rangle$ и $\langle M_b, \nu_b \rangle$ есть множество $\langle M_a \cup M_b, \nu_{a \cup b} \rangle$, где для каждого элемента m из универсума $\nu_{a \cup b}(m) = \max\{\nu_a(m), \nu_b(m)\}$.

На рис. 8.4 приведен пример функции $\nu_{M_1 \cup M_2}$ для множества $M_1 \cup M_2$, которые представлены на рис. 8.1 и рис. 8.2.

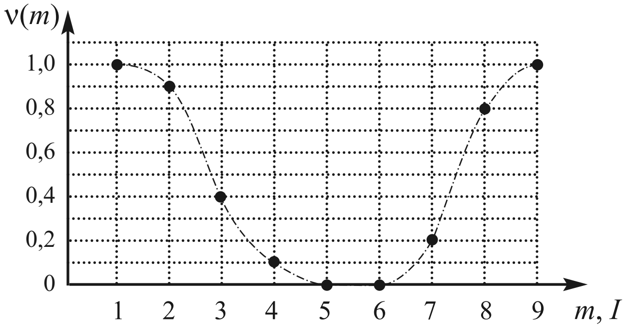


Рис. 8.4. Функция принадлежности для $M_1 \cup M_2$

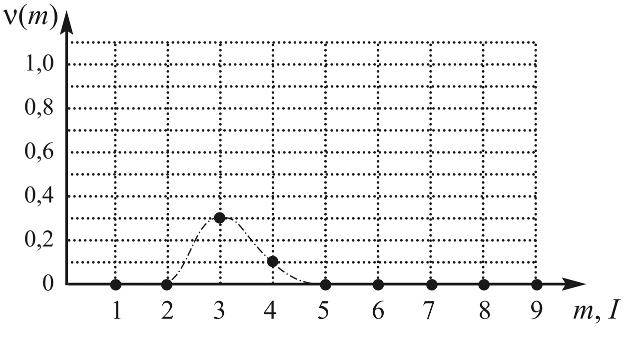


Рис. 8.5. Функция принадлежности для $M_1 \cap M_3$

Пересечение ($\cap$). Пересечение двух нечетких множеств $\langle M_a, \nu_a\rangle$ и $\langle M_b, \nu_b\rangle$ есть множество $\langle M_a \cap M_b, \nu_{a\cap b}\rangle$, где для каждого элемента m из универсума $\nu_{a\cap b}(m) = \min\{\nu_a(m), \nu_b(m)\}$.

На рис. 8.5 приведен пример функции $\nu_{M_1 \cap M_2}$ для множества $M_1 \cap M_3$, которые представлены на рис. 8.1 и рис. 8.3.

Дополнение ($\overline{}$). Дополнение множества M до универсума есть множество $\langle \overline{M}, \overline{\nu}\rangle$, где для каждого элемента m из универсума $\overline{\nu}(m) = 1 - \nu(m)$.

На рис. 8.6 приведен пример функции $\overline{\nu}$ для множества M_1, которое приведено на рис. 8.1.

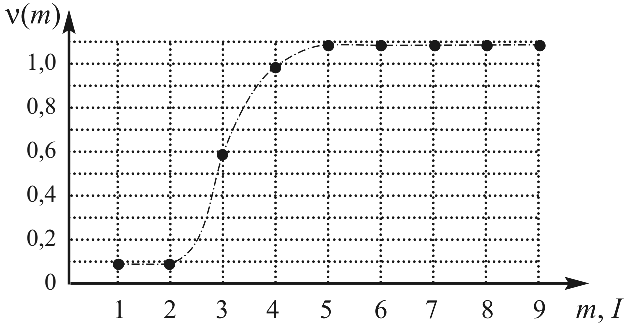


Рис. 8.6. Функция принадлежности для $\overline{M}_1$

Задачи и упражнения

① Вычислить значение меры истинности нечетких составных логических высказываний

$$f_1(a, b, c) = \overline{\overline{\overline{a \vee b} \to \overline{c}} \wedge \overline{\overline{a} \leftrightarrow (b \oplus \overline{c})}};$$

$$f_2(a, b, c) = \overline{\overline{a \vee \overline{b}} \to \overline{\overline{c} \wedge a \leftrightarrow (\overline{b \oplus c})}};$$

$$f_3(a, b, c) = \overline{\overline{\overline{a \vee b} \to \overline{c} \wedge \overline{a}} \leftrightarrow (\overline{b \oplus c})}$$

при следующих возможных значениях μ:

$\mu_a = \{0;\ 0{,}3;\ 0{,}7\}$;

$\mu_b = \{0{,}1;\ 0{,}4\}$;

$\mu_c = \{0{,}5;\ 0{,}8;\ 1\}$.

② На рис. 8.7 графически заданы меры истинности μ трех нечетких логических переменных x_1, x_2, и x_3, определенных на некотором интервале $[0, k]$ параметра p.

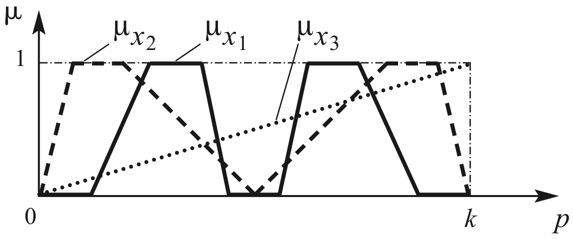


Рис. 8.7. Графики меры истинности $\mu_{x_1}(p)$, $\mu_{x_2}(p)$, $\mu_{x_3}(p)$

Построить графики меры истинности для следующих нечетких логических функций:

μ_{φ_1}, где $\varphi_1 = \varphi_1(x_1) = \bar{x}_1$;

μ_{φ_2}, где $\varphi_2 = \varphi_2(x_2, x_3) = x_2 \wedge \bar{x}_3$;

μ_{φ_3}, где $\varphi_3 = \varphi_3(x_1, x_2, x_3) = \bar{x}_1 \rightarrow x_2 \wedge x_3$;

μ_{φ_4}, где $\varphi_4 = \varphi_4(x_1, x_2, x_3) = \bar{x}_2 \oplus x_3 \vee x_1$;

μ_{φ_5}, где $\varphi_5 = \varphi_5(x_1, x_2, x_3) = \overline{\bar{x}_2 \wedge x_3 \vee x_1}$;

μ_{φ_6}, где $\varphi_6 = \varphi_6(x_1, x_2, x_3) = \overline{x_2 \wedge \overline{x_3 \rightarrow (\bar{x}_1 \rightarrow \bar{x}_2})} \vee x_3$.

③ Показать, что если две функции $f(x_1, x_2, \ldots, x_n)$ и $g(x_1, x_2, \ldots, x_n)$ являются неравносильными, то $\mu(f(x_1, x_2, \ldots, x_n) \leftrightarrow g(x_1, x_2, \ldots, x_n)) \leq 0{,}5$.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Акимов О.Е.* Дискретная математика: логика, группы, графы. – М.: Лаборатория Базовых Знаний, 2001. – 352 с.

2. *Баранов С.И.* Синтез микропрограммных автоматов (граф-схемы и автоматы). 2-е изд., перераб. и доп. – Л.: Энергия, 1979. – 232 с.

3. *Гаврилов Г.П.*, *Сапоженко А.А.* Задачи и упражнения по дискретной математике. – М.: ФИЗМАТЛИТ, 2005. – 416 с.

4. *Глушков В.М.* Синтез цифровых автоматов. – М.: Физматгиз, 1962. – 262 с.

5. *Горбатов В.А.* Фундаментальные основы дискретной математики. Информационная математика. – М.: Наука. Физматлит, 2000. – 544 с.

6. *Заде Л.А.* Понятие лингвистической переменной и его применение к принятию приближенных решений. – М.: Мир, 1976.

7. *Карпов Ю.Г.* Теория автоматов. – СПб.: Питер, 2003. – 208 с.

8. *Клини С.* Математическая логика. – М.: Мир, 1968. – 351 с.

9. Лекции по дискретной математике / Капитонова Ю.В., Кривой С.Л., Летичевский А.А., Луцкий Г.М. – СПб.: БХВ-Петербург, 2004. – 624 с.

10. *Лихтарников Л.М.*, *Сукачева Т.Г.* Математическая логика: Курс лекций. Задачник – практикум и решения. – СПб.: Издательство «Лань», 1999. – 288 с.

11. *Мендельсон Э.* Введение в математическую логику. – М.: Мир, 1988. – 320 с.

12. *Нефедов В.Н.*, *Осипова В.А.* Курс дискретной математики: Учеб. пособие. – М.: Изд-во МАИ, 1992. – 264 с.

13. *Новиков П.С.* Элементы математической логики. 2-е изд. – М.: Наука, 1973. – 400 с.

14. *Новиков Ф.А.* Дискретная математика для программистов. – СПб.: Питер, 2001. – 304 с.

15. *Попов Б.Н.* Цифровые устройства систем приводов летательных аппаратов: Учеб. пособие. – М.: Изд-во МАИ-ПРИНТ, 2008. – 124 с.

16. *Тишин В.В.* Дискретная математика в примерах и задачах. – СПб.: БХВ-Петербург, 2008. – 352 с.

17. *Яблонский С.В.* Введение в дискретную математику. – М.: Наука, 1986. – 384 с.

Учебное издание

Порешин Петр Петрович
Попов Борис Николаевич

ДИСКРЕТНАЯ МАТЕМАТИКА:
МНОЖЕСТВА, ОТНОШЕНИЯ, ЛОГИКА, АВТОМАТЫ

Редакторы: *Т.В. Кособокова, Л.В. Кутукова*
Компьютерная верстка *Е.Э. Качаловой*

Подписано в печать 10.12.2014.
Бум. офсетная. Формат 60х84 1/16.
Усл. печ. л. 10,93. Уч.-изд. л. 11,75. Тираж 200 экз.
Изд. № 604.

Издательство МАИ-ПРИНТ
(МАИ), Волоколамское ш., д. 4, Москва, А-80, ГСП-3 125993

Отпечатано в ФГУП МОКБ «Марс»,
127473, Москва, 1-ый Щемиловский пер., 16