Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Л. Д. Черемисинова

ДИСКРЕТНАЯ МАТЕМАТИКА

Допущено Министерством образования Республики Беларусь в качестве учебного пособия для студентов учреждений высшего образования по направлениям образования «Радиоэлектронная техника», «Вычислительная техника» и специальности «Инженерно-психологическое обеспечение информационных технологий»

Минск БГУИР 2019

УДК 519.1(075.8)
ББК 22.176я73
Ч-46

Р е ц е н з е н т ы:

кафедра высшей математики №1
Белорусского национального технического университета
(протокол №10 от 22 мая 2018 г.);

профессор кафедры математической кибернетики
Белорусского государственного университета
доктор педагогических наук, профессор,
кандидат физико-математических наук О. И. Мельников

Черемисинова, Л. Д.

Ч-46 Дискретная математика : учеб. пособие / Л. Д. Черемисинова. – Минск : БГУИР, 2019. – 299 с. : ил.

ISBN 978-985-543-460-4.

Изложены материалы по основным разделам дискретной математики, описаны комбинаторные алгоритмы на матрицах и графах. Издание основано на лекционном курсе, который читается автором в учреждении образования «Белорусский государственный университет информатики и радиоэлектроники» для студентов специальностей, связанных с вычислительной техникой и программированием.

УДК 519.1(075.8)
ББК 22.176я73

ISBN 978-985-543-460-4

Содержание

ПРЕДИСЛОВИЕ

Дискретная (или конечная) математика является важнейшей составляющей современной математики. В отличие от традиционной математики (алгебры, математического анализа и др.) она изучает дискретные математические объекты и структуры, такие как множества, отношения, графы, логические высказывания, булевы функции. Своеобразие объектов, имеющих дискретный характер, потребовало применения специфических методов их изучения, отличных от методов, применяемых в классической математике, что привело к выделению дискретной математики как отдельного направления.

Несмотря на то что элементы дискретной математики появлялись в течение всего периода развития математики, дискретная математика как наука оформилась и интенсивно развивается только с середины прошлого века, что связано с появлением и внедрением во все сферы науки и производства вычислительной техники и цифровых технологий. Дискретная математика стала основой проектирования и программирования цифровых электронных систем.

Дискретная математика как математическая дисциплина объединила в себе такие фундаментальные разделы, как теория множеств и отношений, графов, алгоритмов, булевы функции и автоматы, математическая логика, представляющие математический аппарат информатики и вычислительной техники. Знание понятий, задач и методов дисциплин, составляющих дискретную математику, совершенно необходимо для специалистов, профессиональная сфера деятельности которых напрямую связана с использованием компьютера как средства решения стоящих перед ними задач. Дискретная математика служит хорошим средством и языком для построения и анализа моделей не только в информатике и программировании, но и в областях, казалось бы, далеких от них: экономике, технике, медицине, социальной сфере, химии, физике и др. Она является основным инструментом при формализации и решении прикладных задач, объекты которых имеют дискретный характер.

Дискретная математика является важнейшим элементом математического образования и обязательным курсом в учебной программе бакалавров по специальностям, связанным с программированием и проектированием. Данное учебное пособие представляет собой введение в дискретную математику как для студентов, не знакомых с ней, так и имеющих понятие об основных проблемах в этой области. Представленные в пособии разделы дискретной математики интересны как сами по себе, так и в связи с тем, что они знакомят с математическим аппаратом, широко применяемым в дисциплинах, связанных с программированием и проектированием вычислительных устройств.

Основная цель данного учебного пособия дать студентам необходимый математический аппарат для успешного решения практических задач в программировании и проектировании дискретных систем, способствовать формированию навыков логического мышления, формализации и разработки алгоритмов решения логических задач.

Настоящее учебное пособие охватывает основные разделы дискретной математики, знание которых необходимо инженерам, специализирующимся в области вычислительной техники: теорию множеств и отношений, математическую логику и теорию булевых функций, комбинаторику и теорию графов. Кроме того, затронуты и некоторые более специальные разделы, необходимые специалистам в области вычислительных устройств, такие как минимизация и схемная реализация булевых функций. За рамками учебного пособия остались такие разделы, часто включаемые в курсы дискретной математики, как кодирование информации, алгебраические структуры, теория чисел.

1 МНОЖЕСТВА

Теория множеств как математическая дисциплина ведет начало от немецкого математика Кантора (1845–1918). Он проводил исследования в области тригонометрических рядов и числовых последовательностей и пришел к задаче сравнения бесконечных множеств по величине. Для решения этой проблемы Кантор ввел понятие мощности множества, считая, по определению, что два множества имеют одинаковую мощность, если каждому элементу одного множества можно поставить в соответствие ровно один элемент другого множества, образовав таким образом пары элементов из сравниваемых множеств. Для конечных множеств такое соответствие можно установить только в том случае, когда они имеют одинаковое число членов. Однако при сравнении по мощности бесконечных множеств это определение также имеет смысл.

К 1890 г. канторовская теория множеств получила признание в качестве самостоятельного раздела математики. Позднее в этой теории были обнаружены противоречия или парадоксы и была разработана более сложная теория, позволившая избежать этих парадоксов. Но прикладные науки до сих пор используют канторовскую трактовку теории множеств – «наивную» теорию.

1.1 Основные понятия

1.1.1 Множества и элементы

Понятие множества является одним из фундаментальных первичных понятий математики. Согласно Кантору *множество* есть любая совокупность определенных и различимых между собой объектов, имеющих что-то общее и мыслимых как единое целое. Эти объекты называются *элементами* множества. Множество состоит из элементов и определяется своими элементами, например: множество студентов некоторого вуза, множество изучаемых студентами радиофизического факультета предметов и т. д. Понятия множества и его элементов являются одними из основных исходных понятий математики, и поэтому точного определения для них нет.

Можно выделить следующие существенные особенности определения множества:

– само множество может рассматриваться также как *единое целое*;

– в обыденной жизни внимание переносится с отдельных объектов на их совокупности, рассматриваемые как один объект (компания, стая, стадо и т. д.);

– элементы множества *различны* и *отличимы* друг от друга;

– элементы множества *не упорядочены*;

– формулировка понятия множества не накладывает *никаких ограничений на природу элементов* (например, множество красных яблок, книг, простых чисел, студентов);

– элементами множества могут быть другие множества (например, множество групп студентов университета).

Множества обычно обозначаются прописными буквами латинского алфавита $A, B, C, \ldots, Z$, элементы – строчными буквами $a, b, c, \ldots, z$.

Если объект a является элементом множества A, то говорят, что a принадлежит множеству A, этот факт обозначается как $a \in A$. Если же некоторый элемент a не принадлежит множеству A, то это обозначается как $a \notin A$. Отношение «$\in$» называется отношением принадлежности.

Множества бывают *конечными* (содержащими конечное число элементов) и *бесконечными*. Примеры бесконечных множеств:

– множество $\mathbb{N}$ натуральных чисел;

– множество $\mathbb{Z}$ целых чисел;

– множество $\mathbb{Q}$ рациональных чисел.

Параметром, характеризующим размер множества, является *мощность множества*. Мощностью конечного множества A, обозначаемой как $|A|$, является число его элементов. Если каждый элемент множества A является также и элементом множества B, то A называется *подмножеством* множества B.

В теории конечных множеств существуют понятия множеств, соответствующие нулю и единице в алгебре чисел.

Множество, не имеющее ни одного элемента, называется *пустым* и обозначается как $\varnothing$, его мощность $|\varnothing| = 0$. Пустое множество конечно и является подмножеством любого другого множества.

При рассмотрении конечных множеств вводится понятие *универсального* множества (*универсума*), состоящего из всех рассматриваемых элементов. Это множество обозначается обычно через U. Как правило, при рассмотрении конкретных множеств речь идет только о подмножествах некоторого фиксированного универсального множества U.

В конкретных практических приложениях в качестве универсума могут использоваться разные множества. Например, в элементарной алгебре универсум состоит из всех действительных чисел. При рассмотрении групп студентов БГУИР за универсальное множество логично принять множество всех студентов БГУИР, при рассмотрении групп студентов факультета ФИТУ за универсум можно принять множество студентов этого факультета.

Множество называется *счетным*, если оно равномощно множеству $\mathbb{N}$ натуральных чисел. Для того чтобы определить, является ли некоторое множество A счетным, необходимо установить взаимно однозначное соответствие между элементами множеств A и $\mathbb{N}$, т. е. пронумеровать элементы из A. Например, множество $\mathbb{Z}$ целых чисел является счетным, так как его элементы можно упорядочить (и перенумеровать) следующим образом 0, 1, –1, 2, –2, … .

Доказано, что:

1) любое подмножество счетного множества конечно или счетно;

2) любое бесконечное множество содержит в себе счетное подмножество;

3) объединение конечного или счетного числа конечных или счетных множеств конечно или счетно.

Из приведенных утверждений следует, что множество $\mathbb{Q}$ рациональных чисел является счетным. Действительно, так как рациональное число можно представить в виде несократимой дроби с целыми числителем и знаменателем, то множество дробей с любым заданным знаменателем счетно. Отсюда следует, что множество $\mathbb{Q}$ можно представить в виде объединения счетного числа счетных множеств, следовательно, оно счетно.

Примером несчетного множества является множество действительных чисел. Мощность этого множества имеет название *континуум*. Континуум является бесконечной мощностью и превосходит мощность счетного множества. Множество, имеющее мощность континуум, называется *континуальным* множеством. Континуальными являются, например, множества всех точек отрезка [0, 1] и всех подмножеств любого счетного множества.

1.1.2 Способы задания множеств

Для того чтобы задать множество, необходимо указать, какие элементы ему принадлежат. Одно и то же множество часто может быть задано разными способами.

1. *Перечисление элементов*. Это простейший способ задания конечного множества. Если множество A состоит, например, из элементов $a_1, a_2, \ldots , a_n$, то его можно задать перечислением этих элементов как $A = \{a_1, a_2, \ldots , a_n\}$. Порядок следования элементов в множестве не является существенным. $A = \varnothing$ или $A = \{\ \}$ задает пустое множество A.

2. *Указание отличительного признака элементов*. При таком способе задается одно или несколько свойств, по которым определяется принадлежность элементов к данному множеству. Если $P(a)$ задает некоторое условие, определяющее, что a обладает свойством P, то $A = \{a / P(a)\}$ есть множество всех тех

и только тех элементов, которые обладают свойством P. Например, $A = \{a \,/\, a \in \mathbb{N},\ a < 5\}$ – множество натуральных чисел, меньших 5. Этот способ применим для задания не только конечных, но и бесконечных множеств. Например, $A = \{a \,/\, a = 2n,\ n \in \mathbb{N}\}$ задает множество четных натуральных чисел.

3. *Индуктивный* (*рекурсивный*) *способ*. Задается некоторая порождающая процедура, которая определяет способ получения элементов множества из уже известных элементов. Этот способ также применим для задания как конечных, так и бесконечных множеств. Например, для бесконечного множества $A = \{1, 3, 5, 7, 9, \ldots\}$ такой порождающей процедурой является следующая:

$$1)\ 1 \in A;$$

$$2)\ \text{если } a \in A, \text{ то } a + 2 \in A.$$

4. *Алгебраический способ*. Задается формула получения множества, исходя из известных множеств, с помощью алгебраических операций над ними. Например, $M = A \cup B$. Здесь множество M содержит все элементы, входящие в хотя бы в одно из множеств A и B.

5. *Визуальное представление множеств*. Множества изображаются на плоскости в виде фигур, называемых *диаграммами Эйлера – Венна*. Обычно на этих диаграммах универсальное множество U представляется частью плоскости, ограниченной замкнутой кривой. Элементам множества U соответствуют точки, находящиеся внутри полученной фигуры, например прямоугольника. Множества представляются кругами внутри этого прямоугольника. Пример диаграммы Эйлера – Венна приведен на рисунке 1.1.

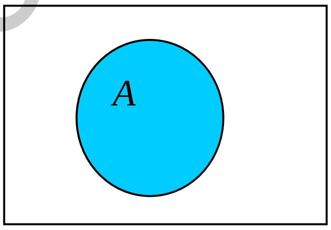


Рисунок 1.1 – Визуальное представление множества A

Этот способ используется обычно только для наглядной демонстрации операций над множествами или отношений между множествами.

6. *Векторное представление множеств*. Если привлечь понятие универсального множества U, тогда всякое множество A, подлежащее рассмотрению, считается его подмножеством. Его можно представить булевым вектором (вектором, компоненты которого равны 0 или 1), число компонент которого равно мощности множества U. Если вектор задает множество A, то его i-я компонента равна 1, если i-й элемент множества U принадлежит множеству A, и 0 – в про-

тивном случае. Например, пусть $U = \{a, b, c, d, e, f, g\}$ и $A = \{a, c, d, f\}$, тогда A представится вектором 1 0 1 1 0 1 0, при этом предполагается, что элементы универсального множества U упорядочены. Векторы 0 0 0 0 0 0 0 и 1 1 1 1 1 1 1 задают соответственно пустое множество $\varnothing$ и само универсальное множество U.

Выбор способа представления множества часто зависит от специфики рассматриваемой задачи и способов ее решения. Например, векторное задание множеств удобно для их представления в памяти компьютера, визуальное – для демонстрации отношений между множествами.

1.1.3 Операции над множествами

Как было сказано выше, множество можно представить в виде формулы как результат операций над другими множествами. Для задания формулы необходимо определить операции над множествами, причем множества должны быть определены на одном универсуме.

Рассмотрим сначала бинарные операции, которые определены только для двух операндов: множеств A и B.

Объединение множеств A и B представляет собой множество, содержащее те и только те элементы, которые принадлежат хотя бы одному из множеств – A или B:

$$A \cup B = \{x \,/\, x \in A \text{ или } x \in B\}.$$

Пересечением множеств A и B является множество, содержащее те и только те элементы, каждый из которых принадлежит как A, так и B:

$$A \cap B = \{x \,/\, x \in A \text{ и } x \in B\}.$$

Разность множеств A и B состоит из элементов множества A, которые не принадлежат множеству B:

$$A \setminus B = \{x \,/\, x \in A \text{ и } x \notin B\}.$$

Сумма множеств A и B (или *симметрическая разность* множеств A и B) содержит все элементы из A, не принадлежащие B, и все элементы из B, не принадлежащие A:

$$A + B = \{x \,/\, (x \in A \text{ и } x \notin B) \text{ или } (x \in B \text{ и } x \notin A)\}.$$

Дополнение множества A (в универсуме U) состоит из элементов универсального множества U, не принадлежащих A:

$$\overline{A} = \{x \,/\, x \in U \text{ и } x \notin A\}.$$

Операции над множествами для наглядности можно проиллюстрировать на диаграммах Эйлера – Венна. На рисунке 1.2 затемненными областями на диаграммах показаны результаты выполнения перечисленных операций над множествами.

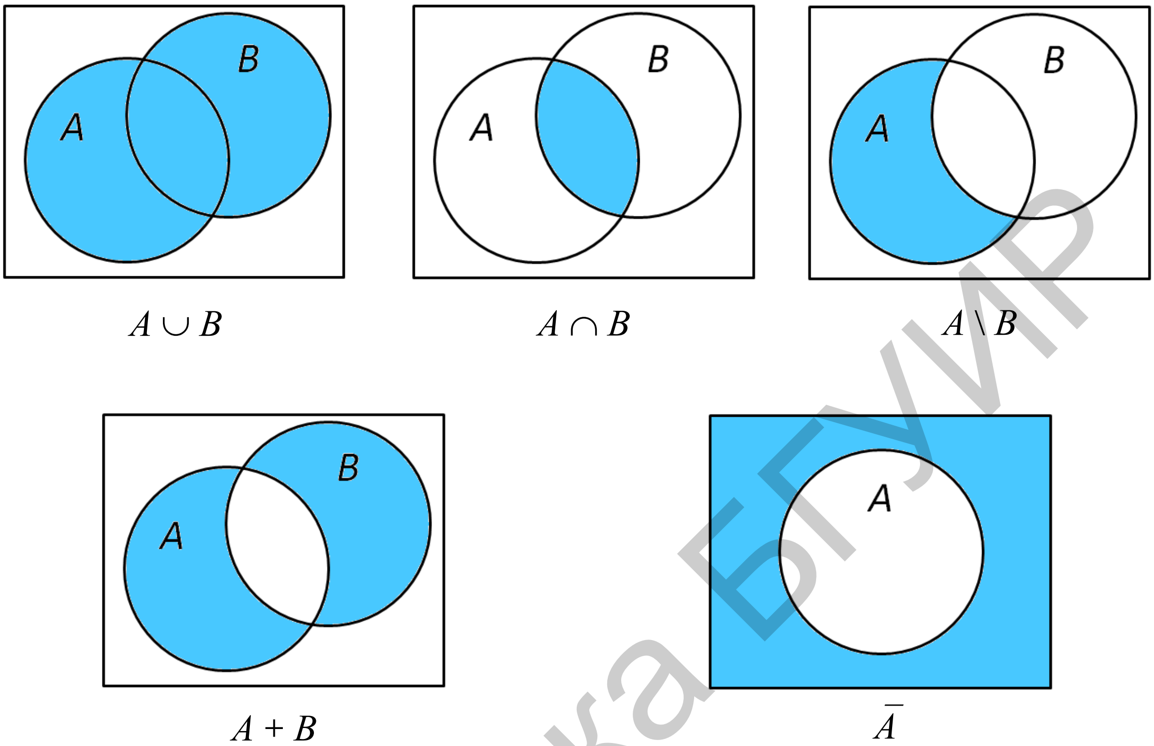


Рисунок 1.2 – Иллюстрация операций над множествами

Бинарные операции объединения и пересечения допускают обобщение на число множеств, большее чем два. Пусть множества A_i, $i = 1, 2,\dots, n$ принадлежат одному и тому же универсальному множеству U ($A_i \subseteq U$), тогда

$$\bigcup_{n}^{i=1} A_i = A_1 \cup A_1 \cup \dots \cup A_n, \quad \bigcap_{n}^{i=1} A_i = A_1 \cap A_1 \cap \dots \cap A_n.$$

Например, пусть $U = \{0, 1, 2, 3\}$, $A = \{0, 1, 2\}$ и $B = \{1, 2, 3\}$. Тогда

$$A \cup B = \{0, 1, 2, 3\}; \;\; A \cap B = \{1, 2\}; \;\; A \setminus B = \{0\}; \;\; A + B = \{0, 3\}; \;\; \bar{A} = \{3\}.$$

1.2 Отношения между множествами

1.2.1 Отношения равенства и включения

Существует два основных отношения между множествами (определенными на одном универсуме): равенство и включение. Пусть A и B – множества.

1. *Равенство*. Множества A и B равны (обозначается $A = B$), если (и только если) они состоят из одних и тех же элементов. Другими словами, каждый

элемент множества A есть элемент множества B, а каждый элемент множества B есть элемент множества A.

Согласно этому определению равными множествами являются множества, различающиеся только порядком перечисления элементов, например, $A = \{0, 1, 2\}$ и $B = \{1, 0, 2\}$.

Неравенство множеств обозначается как $A \neq B$.

2. *Включение*. Множество A является *подмножеством* множества B, если всякий элемент из A принадлежит множеству B. Этот факт обозначается как $A \subseteq B$, где $\subseteq$ – знак включения. При этом говорят, что множество B *содержит*, или *покрывает*, множество A.

Если $A \subseteq B$ и $B \subseteq A$, то множества A и B равны ($A = B$). Эти условия иногда используются в качестве определения отношения равенства между множествами: множества A и B равны ($A = B$), если и только если $A \subseteq B$ и $B \subseteq A$.

Если множество A не является подмножеством множества B, то это обозначается как $A \not\subseteq B$.

Если некоторое непустое множество A является подмножеством множества B ($A \subseteq B$) и $A \neq B$, то говорят, что A является *собственным подмножеством* множества B. Этот факт обозначается как $A \subset B$, где $\subset$ – знак строгого включения в отличие от знака $\subseteq$ нестрогого включения.

Пустое множество, а также само A являются, по определению, *несобственными подмножествами* множества A.

На диаграмме Эйлера – Венна включение одного множества A в B изображается (рисунок 1.3) в виде вложения одной области в другую.

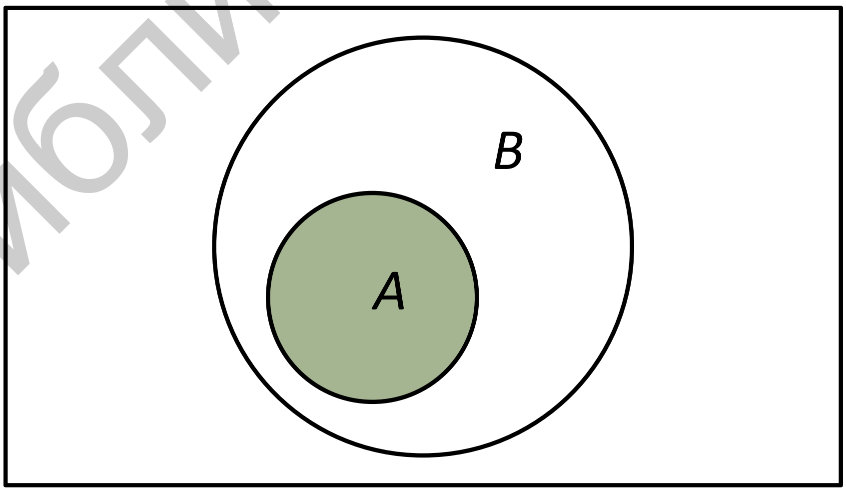


Рисунок 1.3 – Включение множества A в множество B

Например,

– если A – множество всех людей, B – мужчин, то $B \subset A$;

– множество четных чисел является собственным подмножеством множества натуральных чисел;

– множество $A = \{1, 2, 3, 4, 5\}$ является подмножеством (но не собственным) множества $B = \{x / x \in \mathbb{N}, 0 < x < 6\}$.

Кроме упомянутых основных отношений между множествами можно ввести еще одно отношение, не имеющее обозначения – *отношение пересечения*. Два множества не пересекаются, если они не имеют ни одного общего элемента, т. е. $A \cap B = \varnothing$, и пересекаются, если $A \cap B \neq \varnothing$.

Например, пусть $U = \{0, 1, 2, 3\}$, $A = \{0, 1, 2\}$, $B = \{1, 2, 3\}$, $C = \{3, 4\}$. Тогда A и B пересекаются, так как $A \cap B = \{1, 2\} \neq \varnothing$. Но A и C не пересекаются, так как $A \cap C = \varnothing$.

Некоторые свойства подмножеств.

1. Пустое множество является подмножеством любого множества A, т. е. $\varnothing \subseteq A$.

2. Любое множество A является своим несобственным подмножеством: $A \subseteq A$.

3. Если $a_i \in A$ любой элемент из A, то $\{a_i\} \subseteq A$.

4. Всякое множество A, подлежащее рассмотрению, считается подмножеством универсального: $A \subseteq U$.

5. Для любого множества A справедливо $\varnothing \subseteq A \subseteq U$.

6. Для любой пары множеств A и B справедливо

$$\varnothing \subseteq A \cap B \subseteq A \subseteq A \cup B \subseteq U.$$

Необходимо различать и правильно использовать знаки $\in$ и $\subseteq$. Первый знак задает отношение между разными объектами: множествами и элементами множеств, второй знак – отношение между одинаковыми объектами: множествами на одном универсуме. Например, пусть имеем множество $\mathbb{N}$ натуральных чисел. Каждое отдельное натуральное число $n_i \in \mathbb{N}$, а использование отношения $\subseteq$ возможно только в том случае, если речь идет о множестве, включающем это число n_i, т. е. $\{n_i\} \subseteq \mathbb{N}$.

Можно легко доказать, что для любых множеств A, B и C, таких, что $A \subseteq B$ и $B \subseteq C$, справедливо, что $A \subseteq C$. Но несправедливо, что если $A \in B$, $B \in C$, то $A \in C$.

1.2.2 Булеан

Множество всех подмножеств некоторого множества A называется *булеаном*. Булеан обозначается как 2^A или $P(A)$. Среди его элементов находится и само множество A, а также пустое множество $\varnothing$.

Например, булеаном множества $A = \{a, b, c\}$ является множество

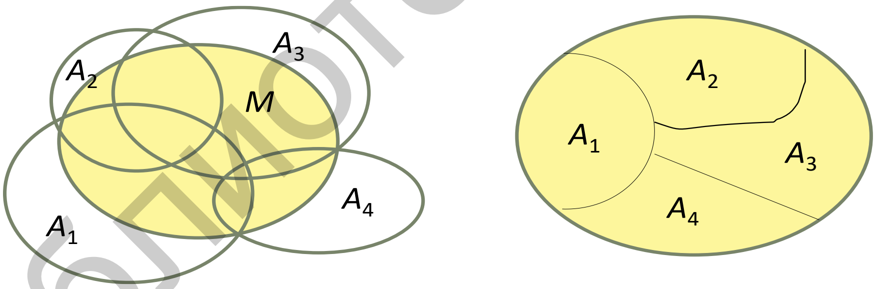
$$2^A = \{\varnothing, \{a\},\{b\},\{c\},\{a, b\},\{a, c\},\{b, c\},\{a, b, c\}\}.$$

Мощность булеана множества A равна $2^{|A|}$. Действительно, $2^{\varnothing} = \{\varnothing\}$, т. е. число элементов булеана пустого множества есть $|2^0| = 1$, а добавление к A одного нового элемента каждый раз увеличивает мощность его булеана вдвое. В частности, булеан пустого множества A есть $P(\varnothing) = \{\varnothing\}$ или, что то же, $P(\varnothing) = \{\{\}\}$, булеан одноэлементного множества $A = \{a\}$ есть $P(\{a\}) = \{\{\},\{a\}\}$.

1.2.3 Разбиение и покрытие множеств

Пусть $A = \{A_1, A_2,..., A_n\}$ семейство непустых подмножеств множества U: $A_i \subset U$. Семейство A подмножеств является *покрытием* некоторого множества M, если их объединение содержит множество M, т. е. каждый элемент множества M принадлежит хотя бы одному A_i (рисунок 1.4, *а*):

$$M \subseteq A_1 \cup A_2 \cup ... \cup A_n.$$



а *б*

Рисунок 1.4 – Покрытие (*а*) множества M множествами A_1, A_2, A_3, A_4
и разбиение (*б*) множества M

В определении понятия покрытия множества существенно то, что:

– все подмножества A_i не равны пустому множеству, т. е. обязательно должны содержать хотя бы один элемент;

– подмножества A_i могут пересекаться, т. е. некоторый элемент из M может входить одновременно в два подмножества;

– объединение всех подмножеств должно содержать в себе исходное множество M.

Разбиением множества M является представление его в виде объединения произвольного числа попарно непересекающихся непустых подмножеств. Это означает, что семейство A подмножеств является *разбиением* множества M, если (рисунок 1.4, *б*):

1) $M = A_1 \cup A_2 \cup ... \cup A_n$;

2) $A_i \cap A_j = \varnothing$ для всех i, j ($i \neq j$).

В этом случае говорят, что множество M *разбито* на подмножества A_1, A_2,..., A_n, а подмножества A_i называют *блоками разбиения* M.

В определении понятия разбиения множества существенно то, что:

– все подмножества A_i не равны пустому множеству, т. е. обязательно должны содержать элементы;

– подмножества A_i не должны пересекаться, т. е. ни один элемент из M не входит одновременно в два подмножества;

– объединение всех подмножеств должно быть равно исходному множеству M.

Таким образом, разбиением множества M является множество непустых и взаимно непересекающихся его собственных подмножеств, называемых блоками.

К операциям разбиения множеств сводится ряд прикладных задач. Например, разбиение множества конструктивных элементов высокого уровня (микросхем, блоков) на элементы более низкого уровня (вентили, субблоки).

Например, пусть $A = \{0, 1, 2, 3\}$, $A_1 = \{0, 1, 2\}$ и $A_2 = \{3\}$:

$$A_1 \cup A_2 = A \text{ и } A_1 \cap A_2 = \varnothing, \text{ отсюда } A \text{ разбито на } A_1, A_2.$$

1.3 Булева алгебра множеств

1.3.1 Формулы над множествами

Используя операции над множествами, можно получать формулы. Математически строго *формула* определяется индуктивно:

1. Символы множеств $\varnothing$, U, A, B, C,… есть формулы.

2. Если P и Q формулы, то следующие выражения также являются формулами:

$$(\overline{P}), (P \cup Q), (P \cap Q), (P \setminus Q), (P + Q).$$

3. Все формулы могут быть получены путем итеративного применения 1 и 2.

Две формулы *равносильны*, если они представляют одно и то же множество. Некоторые операции над множествами можно выразить через другие. Так, например,

$$A + B = (A \cap \overline{B}) \cup (\overline{A} \cap B) = (A \cup B) \setminus (A \cap B);$$

$$\overline{A} = U \setminus A;$$

$$A \setminus B = A \cap \overline{B}.$$

Скобки в формулах могут опускаться, если это не приводит к двусмысленности. При этом используется общепринятое правило: если в формуле отсутствуют скобки, устанавливающие порядок выполнения операций, то сначала выполняется дополнение, потом пересечение, а затем объединение, сумма и разность, т. е. последние три операции имеют один приоритет. Символ операции пересечения в формулах часто опускают для повышения наглядности и компактности представления. Например,

$$(A \cap \overline{B}) \cup (\overline{A} \cap B) = A\,\overline{B} \cup \overline{A}\,B.$$

1.3.2 Основные законы алгебры множеств

Алгебра множеств – теоретико-множественный аналог алгебры действительных чисел. В общем случае *алгебра* – множество объектов и операций над ними, подчиняющихся законам – тождествам (справедливым при всех значениях входящих в них объектов).

Три операции теории множеств: дополнение, пересечение и объединение, составляют *булеву алгебру множеств* на некотором универсуме U. Перечислим основные законы этой алгебры

Коммутативность:

$A \cup B = B \cup A;$ $\qquad\qquad$ $A \cap B = B \cap A.$

Ассоциативность:

$A \cup (B \cup C) = (A \cup B) \cup C;$ $\qquad\qquad$ $A\,(B \cap C) = (A \cap B) \cap C.$

Дистрибутивность (распределительность, одной операции относительно другой):

$A \cap (B \cup C) = A \cap B \cup A \cap C;$ $\qquad\qquad$ $A \cup B \cap C = (A \cup B) \cap (A \cup C).$

Идемпотентность (свойство операции при повторном ее применении к объекту давать тот же результат, термин получен путем комбинации двух латинских слов: «idem» («тот же самый») и «potens» («способный»)):

$A \cup A = A;$ $\qquad\qquad$ $A \cap A = A.$

Законы де Моргана (отрицание конъюнкции есть не что иное, как дизъюнкция отрицаний, и наоборот):

$\overline{A \cup B} = \overline{A} \cap \overline{B}$; $\overline{AB} = \overline{A} \cup \overline{B}$.

Законы операций с константами (пустым и универсальным множествами):

$A \cup \varnothing = A$; $A \cap \boldsymbol{U} = A$;

$A \cup \boldsymbol{U} = \boldsymbol{U}$; $A \cap \varnothing = \varnothing$;

$A \cup \overline{A} = \boldsymbol{U}$; $A \cap \overline{A} = \varnothing$.

Закон двойного дополнения:

$\overline{\overline{A}} = A$.

Можно заметить, что для каждой пары формул, представляющих тот или иной закон, справедлив *принцип двойственности*, заключающийся в том, что одна из формул получается из другой взаимной заменой операций пересечения на операции объединения и символов $\varnothing$ на символы $\boldsymbol{U}$.

Например,

$$A \cup B = B \cup A \text{ и } A \cap B = B \cap A;$$

$$A \cup \varnothing = A \text{ и } A \cap \boldsymbol{U} = A.$$

1.3.3 Равносильные преобразования в алгебре множеств

В булевой алгебре множеств любое равенство можно вывести путем равносильных преобразований, используя основные законы этой алгебры.

Приведем примеры вывода некоторых интересных тождеств булевой алгебры множеств.

1. *Поглощение* $A \cup A\,B = A$:

$$A \cup A\,B = A\,\boldsymbol{U} \cup A\,B = A\,(\boldsymbol{U} \cup B) = A\,\boldsymbol{U} = A.$$

2. *Простое склеивание* $A\,B \cup \overline{A}\,B = B$:

$$A\,B \cup \overline{A}\,B = B\,A \cup B\,\overline{A} = B\,(A \cup \overline{A}) = B\,\boldsymbol{U} = B.$$

3. *Обобщенное склеивание* $A\,B \cup \overline{A}\,C = A\,B \cup \overline{A}\,C \cup B\,C$:

$$A\,B \cup \overline{A}\,C \cup B\,C = A\,B \cup \overline{A}\,C \cup B\,C\,(A \cup \overline{A}) =$$

$$(A\,B \cup A\,B\,C) \cup (\overline{A}\,C \cup \overline{A}\,B\,C) =$$

$$= A\,B\,(\boldsymbol{U} \cup C) \cup \overline{A}\,C\,(\boldsymbol{U} \cup B) = A\,B \cup \overline{A}\,C.$$

На основании принципа двойственности из вышеприведенных формул можно получить следующие тождества:

1. Поглощение $A \cap (A \cup B) = A$;

2. Простое склеивание $(A \cup B)\,(\overline{A} \cup B) = B$;

3. Обобщенное склеивание $(A \cup B)\,(\overline{A} \cup C) = (A \cup B)\,(\overline{A} \cup C)\,(B \cup C)$.

Используя выведенные формулы, можно вывести также формулу *сокращения* $A \cup \overline{A}\,B = A \cup B$ (и соответственно двойственную ей $A\,(\overline{A} \cup B) = A\,B$):

$A \cup \overline{A}\,B = A\,U \cup \overline{A}\,B = A\,U \cup \overline{A}\,B \cup U\,B = A \cup \overline{A}\,B \cup B = A \cup B$.

Задания

1. Верно ли, что (ответ обосновать):
$a \in \{\{a, b, c, d\}\}$;
$a \in \{a\}$;
$a \in \{\{a\}\}$;
$\{a\} \in \{\{a, b\}, \{a\}\}$;
$\{a\} \subseteq \{\{a, b\}, \{a\}\}$;
$1 \in C$ и $4 \in C$, если $A = \{1, 2, 3\}$, $B = \{4, 5, 6\}$, $C = \{A, B\}$;
$\{a, b\} = \{\{a, b\}\}$;
$\{1, 2, 3\} \subseteq \{\{1, 2\}, \{2,3\}\}$;
$\{\{1, 2\}\} \subseteq \{\{1, 2\}, \{2,3\}\}$;
$\{1\} \in \mathbb{N}$;
$1 \subseteq \mathbb{N}$;
$\{1\} \subseteq \mathbb{N}$;
$\{1, 2\} \subseteq N$?

2. В каком отношении находятся множества:
$\{\{1, 2\}, \{2, 3\}\}$ и $\{1, 2, 3\}$;
$\varnothing$ и $\{\varnothing\}$;
$A = \{a, b, c\}$ и $B = \{\{a, b\}, \{c\}\}$;
$A = \{a, b, c\}$ и $B = \{a, c, d\}$;
$P(A \cup B)$ и $P(A) \cup P(B)$;
$P(A \cap B)$ и $P(A) \cap P(B)$;
$P(A)$ и $P(A^{*})$, если $A = \{a_1, a_2, a_3\}$, $A^{*} = \{a_1, a_2, a_3, a_4\}$?

3. Сколько элементов содержат множества:
$A = \{a\}$, $B = \{\{a\}\}$; $C = \{\{a, b\}\}$; $D = \{\{a, b\}, \{b, c, d\}\}$?

4. Определить булеан $P(A)$ множеств:
$A = \{0, 1, 2\}$;

$A = \{\{a, b\}, \{c\}\}$;

$A = \{a, b, c\}$;

$A = \{\{a, b\}, \{c\}, \varnothing\}$;

$A = \{0, -1, -2\}$.

5. Доказать тождества (путем равносильных преобразований, используя основные законы алгебры множеств), справедливые для любых $A, B, C \subseteq U$, и проиллюстрировать на диаграмме Эйлера – Венна:

$A \setminus (A \setminus B) = A \cap B$;

$A \cap (B \cup C) = A \cap B \cup A \cap C$ (как называется этот закон ?);

$A \cap (A \cup B) = A$ (как называется этот закон ?);

$A \cup (A \cap B) = A$ (как называется этот закон ?);

$A \cup (B \cap \bar{A}) = A \cup B$;

$A \cap (B \cup \bar{A}) = A \cap B$;

$(A \cup B) \setminus C = (A \setminus C) \cup (B \setminus C)$;

$A \setminus (B \setminus C) = (A \setminus B) \cup (A \cap C)$;

$A \setminus (B \cup C) = (A \setminus B) \setminus C$;

$A \cap (B + C) = A\,B + A\,C$;

$A + B = (A \setminus B) \cup (B \setminus A)$;

$A + B = A\,\bar{B} \cup \bar{A}B$;

$(A \cup B) = A + B + (A \cap B)$;

$A \cap (B + C) = (A \cap B) + (A \cap C)$;

$A \cup B = (A + B) \cup (A \cap B)$.

6. Доказать, что множества $A \cup B$ и $\bar{A} \cap \bar{B}$ не пересекаются. Проиллюстрировать на диаграмме Эйлера – Венна.

7. Выразить операции:

$\cup$ через $\cap$ и $+$;

$\setminus$ через $\cap$ и $+$;

$\cap$ через $\cup$ и $+$;

$\setminus$ через $\cup$ и $+$;

$\cap$ через $\setminus$ и $+$;

$\cup$ через $\setminus$ и $+$;

$+$ через $\cup$, $\cap$ и $\neg$;

$\setminus$ через $\cup$, $\cap$ и $\neg$;

$+$ через $\cup$ и $\setminus$.

8. Существуют ли множества *A, B, C*, для которых имеют место следующие соотношения (пояснить)?

$C \neq \varnothing, \quad A \cap B \neq \varnothing, \quad A \cap C = \varnothing, \quad (A \cap B) \setminus C = \varnothing.$

9. Упростить следующие формулы:

$(A \cap B \cup C) \cup (\bar{A} \cap B \cup C) \cup \bar{B} \cup \bar{C};$

$\neg(\bar{A} \cap (A \cup B) \cap (A + B));$

$(A + \neg((A \cap C) \cup B)) \cup \neg(A \setminus C);$

$(A \cap B \cup C) \cup (\bar{A} \cap B \cup C) \cup \bar{B} \cup \bar{C};$

$(A \cap B \cap D) \cup (A \cap B \cap C \cap D \cap E) \cup (A \cap D \cap \bar{A});$

$\neg(A \cup B) \cup \bar{A};$

$\neg(\bar{A}B \cup B);$

$AB \cup \bar{A}B \cup A\bar{B};$

$(A \cup B)(\bar{A} \cup C)(\bar{B} \cup D)(\bar{C} \cup D);$

$ABD \cup ABCDE \cup AD\bar{A};$

$ABC\bar{D} \cup \bar{A}C \cup \bar{B}C \cup CD.$

10. Верны ли в общем случае следующие утверждения (ответ обосновать)?

Если $A \in B$ и $B \in C$, то $A \in C$;

если $A \subseteq B$ и $B \in C$, то $A \in C$;

если $A \cap B \subseteq \bar{C}$ и $A \cup C \subseteq B$, то $A \cap C = \varnothing$;

если $A \neq B$ и $B \neq C$, то $A \neq C$;

если $A \subseteq \neg(B \cup C)$ и $B \subseteq \neg(A \cup C)$, то $B = \varnothing$.

11. Доказать, что для любых множеств $A, B, C \subseteq U$ имеют место утверждения:

если $A \subset B$ и $C \subset D$, то $(A \cup C) \subset (B \cup D)$;

если $A \subset B$ и $C \subset D$, то $(A \cap C) \subset (B \cap D)$;

если $A \cup B \subseteq \bar{C}$ и $A \cup C \subseteq B$, то $A \cap C = \varnothing$;

если $A \cup B \subseteq \bar{C}$, то $A \subseteq \bar{B} \cup C$, и если $A \subseteq \bar{B} \cup C$, то $A \cup B \subseteq \bar{C}$;

если $(A \setminus B) \cup B = A$, то $B \subseteq A$, и если $B \subseteq A$, то $(A \setminus B) \cup B = A$;

если $A \cup B \subset C$, то $A \subseteq C$ и $B \subseteq C$, и если $B \subseteq C$, то $A \cup B \subset C$.

12. Найти, используя диаграммы Эйлера – Венна, корень *X* систем уравнений):

$A \cap X = B$ и $A \cup X = C$, если $B \subseteq A \subseteq C$;

$X \setminus A = C$ и $A \setminus X = B$, если $B \subseteq A, A \cap C = \varnothing$;

$A \setminus X = B$ и $A \cup X = C$, если $B \subseteq A \subseteq C$.

13. Доказать (и проиллюстрировать на диаграмме Эйлера – Венна), что

$|A \cup B| = |A| + |B| - |A \cap B|$.

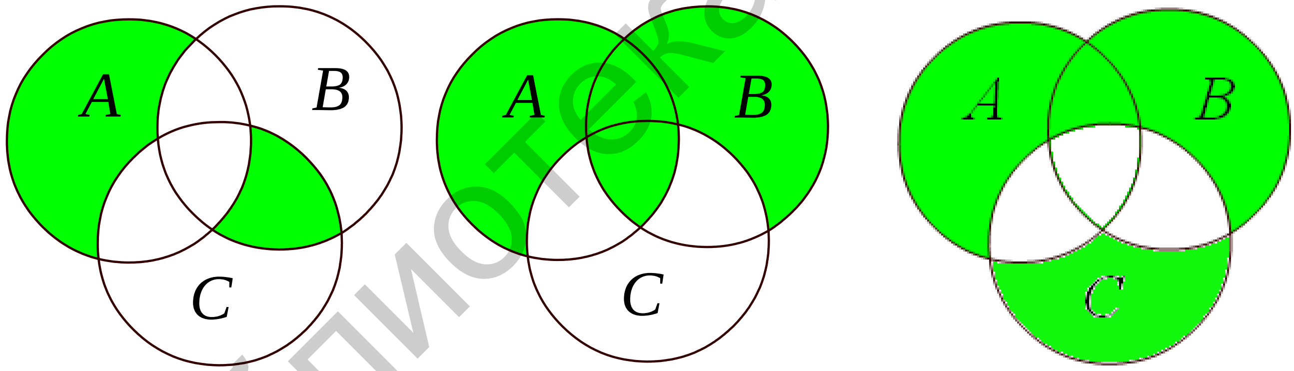
Обобщить на случай трех множеств: $|A \cup B \cup C|$.

14. Представить следующие формулы в базисе: а) пересечение, объединение, дополнение; б) пересечение, дополнение; в) объединение, дополнение:

$(B + \bar{C}) \setminus (C \cap \bar{A} \cap \bar{B})$;
$(B + \bar{C}) \setminus (\bar{A} \cap \bar{B} \cap \bar{C})$;
$(B \cap \bar{C}) \cup \neg(A \cap \bar{B} \cap \bar{C})$.
Упростить и показать на диаграмме Эйлера – Венна.

15. Исследовать с помощью диаграмм Эйлера – Венна вопрос о справедливости следующего утверждения: «Если для множеств A, B и C (определенных на одном и том же универсуме), имеет место $A \cap B \subset \bar{C}$ и $A \cup C \subset B$, то $A \cap C = \varnothing$».

16. Задать множества, представленные закрашенными областями диаграмм Эйлера – Венна, в алгебраическом виде:



2 ОТНОШЕНИЯ

Отношения лежат в основе построения подавляющего большинства математических моделей дискретной математики, используемых для решения практических задач. Отношение в математике – математическая структура, которая формально определяет свойства различных объектов и их взаимосвязи. Распространенными примерами отношений в математике являются равенство, делимость, подобие, параллельность и др. В теории множеств понятие отношения формализовано: отношение определяется через понятие декартова произведения множеств.

2.1 Декартово произведение

Упорядоченная совокупность элементов множества A называется *кортежем* или *вектором* и обозначается как $(a_1, a_2, \dots , a_n)$ или $<a_1, a_2, \dots , a_n>$, где a_i – i-я координата, а n – длина (или размерность) кортежа. Кортеж длиной два представляет собой упорядоченную пару (a, b) (или $<a, b>$), длиной три – упорядоченную тройку (a, b, c), длиной n – упорядоченную n-ку $(a_1, a_2, \dots , a_n)$. Кортежи $(a_1, a_2, \dots , a_n)$ и $(b_1, b_2, \dots , b_m)$ равны, если равны их длины и одноименные элементы: $n = m$ и $a_i = b_i$ для всех $i \in \{1, 2, \dots, n\}$.

Например, кортежи (1, 3, 5, 6) и (1, 3, 5), (1, 3, 5, 6) и (3, 5, 1, 6) не являются равными.

Декартовым, или *прямым* (*картезианским*), *произведением* двух множеств A и B, обозначаемым как $A \times B$, называется множество всевозможных упорядоченных пар (a, b), таких, что $a \in A$ и $b \in B$:

$$A \times B = \{(a, b)/a \in A, b \in B\}.$$

Например, если $A = \{a, b, c\}$ и $B = \{1, 2\}$, то $A \times B = \{(a, 1), (b, 1), (c, 1), (a, 2), (b, 2), (c, 2)\}$.

Для конечных множеств A и B, $|A| = n$, $|B| = m$, элементы множества $A \times B$ заполняют все клетки таблицы размером $n \times m$.

Понятие декартова произведения распространяется на случай с более чем двумя множествами. Декартово произведение n множеств $A_1, A_2, \dots , A_n$ обозначается $A_1 \times A_2 \times \dots \times A_n$ и представляет собой множество всевозможных кортежей $(a_1, a_2, \dots , a_n)$ размерностью n, таких, что $a_1 \in A_1$, $a_2 \in A_2$, … , $a_n \in A_n$:

$$A_1 \times A_2 \times \dots \times A_n = \{(a_1, a_2, \dots , a_n)/a_1 \in A_1, a_2 \in A_2, \dots , a_n \in A_n\}.$$

Мощность декартова произведения множеств равна произведению их мощностей: $|A_1 \times A_2 \times \ldots \times A_n| = |A_1| \times |A_2| \times \ldots \times |A_n|$.

Декартово произведение n одинаковых сомножителей $A \times A \times \ldots \times A$ обозначается как A^n и называется n-й *степенью* множества A. При этом $A^1 = A$.

Примеры декартовых произведений.

1. 64 клетки шахматной доски задаются декартовым произведением множеств $A = \{a, b, \ldots , h\}$ и $B = \{1, 2, \ldots , 8\}$: $A \times B = \{(a, 1), (a, 2), \ldots, (h, 7), (h, 8)\}$.

2. Пусть R множество точек прямой линии. Тогда $R^2 = R \times R$ – множество точек на плоскости. Здесь элементы пар $(x, y) \in R^2$ ($x \in R$ и $y \in R$) служат координатами некоторой точки на плоскости. Другим примером является множество R^3 точек в трехмерном евклидовом пространстве. Обобщением этих понятий является n-мерное пространство.

3. Пусть $A = \{a/1 \le a \le 2\}$ и $B = \{b/0 \le b \le 1\}$ – множества действительных чисел, тогда декартовы произведения $A \times B = \{(a, b)/\ 1 \le a \le 2,\ 0 \le b \le 1\}$ и $B \times A = \{(b, a)/\ 0 \le b \le 1,\ 1 \le a \le 2\}$ задают точки квадратных областей точек на плоскости (рисунок 2.1).

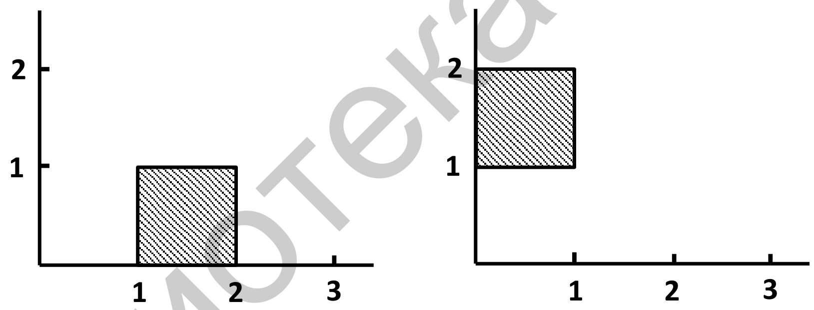


Рисунок 2.1 – Области, задаваемые декартовыми произведениями $A \times B$ и $B \times A$

Последний пример наглядно показывает, что бинарное декартово произведение не коммутативно: $A \times B \neq B \times A$. Но можно показать, что декартово произведение дистрибутивно относительно объединения и пересечения. Если A, B и С множества, то

1) $A \times (B \cup C) = (A \times B) \cup (A \times C)$;

2) $A \times (B \cap C) = (A \times B) \cap (A \times C)$;

3) $(A \cup B) \times C = (A \times C) \cup (B \times C)$;

4) $(A \cap B) \times C = (A \times C) \cap (B \times C)$.

Для примера докажем первое из этих утверждений. Пусть (x, y) – произвольный элемент из $A \times (B \cup C)$, тогда имеет место следующая цепочка утверждений:

$$(x, y) \in A \times (B \cup C) \Leftrightarrow (x \in A) \text{ и } (y \in (B \cup C)) \Leftrightarrow$$

$$\Leftrightarrow x \in A \text{ и } ((y \in B) \text{ или } (y \in C)) \Leftrightarrow$$

$$\Leftrightarrow ((x \in A) \text{ и } (y \in B)) \text{ или } ((x \in A) \text{ и } (y \in C)) \Leftrightarrow$$

$$\Leftrightarrow ((x, y) \in A \times B) \text{ или } ((x, y) \in A \times C) \Leftrightarrow ((x, y) \in (A \times B) \cup (A \times C).$$

2.2 Отношения *n*-арные и бинарные

Подмножество $R \subseteq A_1 \times A_2 \times \ldots \times A_n$ декартова произведения n множеств называется *n-арным отношением*. Если $(a_1, a_2, \ldots, a_n) \in R$, где $a_1 \in A_1$, $a_2 \in A_2$, …, $a_n \in A_n$, то говорят, что элементы $a_1, a_2, \ldots, a_n$ находятся в отношении R.

Отношение $R = \varnothing$ называется пустым. Множество всех кортежей $(a_1, a_2, \ldots, a_n) \in A_1 \times A_2 \times \ldots \times A_n$ может рассматриваться как *универсум* (область определения) для задания отношений $R \subseteq A_1 \times A_2 \times \ldots \times A_n$.

В зависимости от числа n множеств декартового произведения $n = 1, 2, 3$ различают отношения соответственно *унарные*, *бинарные*, *тернарные*.

Унарное отношение на множестве A представляет собой подмножество элементов множества A, обладающих некоторым признаком. Например, подмножество R студентов-отличников или унарное отношение «быть простым числом» на множестве натуральных чисел.

Примерами бинарных отношений являются отношения «<» и «>» на множестве натуральных чисел; отношения параллельности и перпендикулярности на множестве прямых на декартовой плоскости. Отношения «отец – сын» и «отец – мать – ребенок» являются примерами бинарного и тернарного отношений на множестве людей.

Бинарные отношения (или соответствия) образуют наиболее важный и широко используемый в дискретной математике класс отношений.

Бинарным отношением между элементами множеств A и B называется любое подмножество $R \subseteq A \times B$ декартова произведения этих множеств. Тот факт, что некоторый элемент $a \in A$ находится в отношении R с элементом $b \in B$, обозначают как $(a, b) \in R$ или более кратко, в инфиксной форме записи, как $a\,R\,b$.

Пусть $A = \{2, 3\}$ и $B = \{1, 2, 3, 4, 5, 6\}$. В качестве примера бинарного отношения рассмотрим отношение R между элементами множеств «$a \in A$ есть делитель $b \in B$». Тогда $R = \{(2, 2), (2, 4), (2, 6), (3, 3), (3, 6)\}$.

2.2.1 Представление бинарных отношений

В теории отношений используются следующие основные типы представления бинарных отношений.

1. *Перечисление элементов*. Отношение R между элементами множеств A и B задается перечислением тех пар $(a_i, b_l) \in A \times B$, которые принадлежат R. Например, отношение $R = \{(a_1, b_1), (a_1, b_3), (a_2, b_1), (a_2, b_3), (a_2, b_4), (a_3 b_1), (a_3, b_4), (a_5, b_4)\}$ между элементами множеств $A = \{a_1, a_2, a_3, a_4, a_5\}$ и $B = \{b_1, b_2, b_3, b_4\}$.

2. *Матричное представление*. Бинарное отношение представляется в виде *булевой* (двоичной) матрицы. При этом предполагается, что элементы множеств A и B пронумерованы. Элемент булевой матрицы, расположенный на пересечении i-й строки и j-го столбца, имеет значение 1, если i-й элемент $a_i \in A$ находится в отношении R с j-м элементом $b_j \in B$ $(a_i\ R\ b_j)$, в противном случае этот элемент матрицы имеет значение 0.

Например, вышеприведенное отношение R на $A \times B$ представляется следующей булевой матрицей:

$$\boldsymbol{R} = \begin{array}{ccccc} b_1 & b_2 & b_3 & b_4 & \\ 1 & 0 & 1 & 0 & a_1 \\ 1 & 0 & 1 & 1 & a_2 \\ 1 & 0 & 0 & 1 & a_3 \\ 0 & 0 & 0 & 0 & a_4 \\ 0 & 0 & 0 & 1 & a_5 \end{array}$$

3. *Графическое представление*. В графическом виде бинарное отношение представляется направленным двудольным графом $G = (V_1, V_2, E)$ (см. пункт 4.1.3), где V_1 и V_2 – множества вершин первой и второй долей графа, E – множество дуг графа, каждая из которых связывает некоторую вершину первой доли с вершиной второй доли. Вершины первой доли графа G, задающего отношение R, ставятся в соответствие элементам множества A, а вершины второй доли – элементам множества B. Если $a_i\ R\ b_j$, то соответствующие вершины графа связываются дугой $e_{ij} \in E$. На рисунке 2.2 приведено графическое представление вышеприведенного отношения $R \subseteq A \times B$.

Проекцией элемента $(a, b) \in A \times B$ на множество A является элемент a: пр$_A(a, b) = a$. Аналогично элемент b является проекцией элемента (a, b) на множество B: пр$_B(a, b) = b$.

Проекцией множества $T \subset A \times B$ на множество A называется множество всех тех элементов из A, которые являются проекциями элементов из T на множество A:

$$\text{пр}_A T = \{\text{пр}_A(a_i, b_j) \mid (a_i, b_j) \in T\},\ \text{пр}_B T = \{\text{пр}_B(a_i, b_j) \mid (a_i, b_j) \in T\}.$$

Например, для множеств $A = \{a_1, a_2, a_3, a_4, a_5\}$ и $B = \{b_1, b_2, b_3, b_4\}$ проекциями элемента (a_1, b_3) на множества A и B являются a_1 и b_3 ($\text{пр}_A(a_1, b_3) = a_1$, $\text{пр}_B(a_1, b_3) = b_3$), а проекция множества $T = \{(a_2, b_3), (a_3 b_1), (a_3, b_4)\}$ на A – $\text{пр}_A T =$ $= \{a_2, a_3\}$.

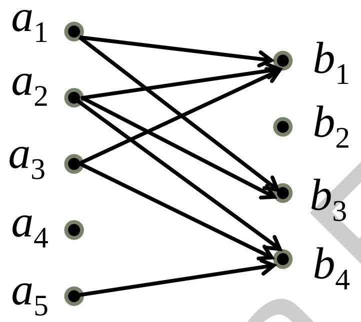


Рисунок 2.2 – Графическое представление отношения $R \subseteq A \times B$

Образом элемента $a \in A$ относительно отношения $R \subseteq A \times B$ называется множество всех тех элементов $b \in B$, которые находятся в отношении R с элементом a: $R(a) = \{b \mid b \in B,\ (a, b) \in R\}$. Образ $R(a_i)$ включает все элементы $b_j \in B$, которым соответствуют единицы i-й строки матричного задания отношения R. В графическом виде им соответствуют концевые вершины всех исходящих из a_i дуг. Например, $R(a_1) = \{b_1, b_3\}$.

Образом подмножества $T \subseteq A$ относительно R называется объединение образов для всех элементов из подмножества T: $R(T) = \{b \mid b \in B,\ x \in T,$ $(x, b) \in R\}$. Например, в случае последнего рассмотренного примера отношения R образом множества $T = \{a_1, a_3\}$ относительно R является $R(T) = \{b_1, b_3, b_4\}$.

Бинарное отношение $R \subseteq A \times B$ можно задавать с помощью образов элементов $a \in A$. Множество образов для всех $a \in A$ называется *фактор-множеством* отношения R. Фактор-множество рассмотренного отношения R есть $\{\{b_1, b_3\}, \{b_1, b_3, b_4\}, \{b_1, b_4\}, \varnothing, \{b_4\}\}$.

Прообразом элемента $b \in B$ и подмножества $Y \subseteq B$ относительно R называются соответственно множества $R^{-1}(b) = \{a \mid a \in A,\ (a, b) \in R\}$ и $R^{-1}(Y) =$ $= \{a \mid a \in A, y \in Y, (a, y) \in R\}$.

Например, для рассмотренного выше примера отношения R прообразом множества $Y = \{b_3, b_4\}$ является $R^{-1}(Y) = \{a_1, a_2, a_3, a_5\}$, а прообраз $R^{-1}(b_i)$ включает все a_j, которым соответствуют единицы i-го столбца матричного задания отношения R. В графическом виде им соответствуют начальные вершины всех входящих в b_i дуг. Например, $R^{-1}(b_1) = \{a_1, a_2, a_3\}$.

Областью определения отношения $R \subseteq A \times B$ является проекция пр$_A R$ множества R на A. Для рассматриваемого выше отношения такой областью является $\{a_1, a_2, a_3, a_5\}$. *Областью значений* отношения $R \subseteq A \times B$ является проекция пр$_B R$ множества R на B. Областью значений рассматриваемого выше отношения R является $\{b_1, b_3, b_4\}$.

Если область определения отношения $R \subseteq A \times B$ совпадает с множеством A (пр$_A R = A$), то отношение R называется *всюду определенным*. Задающая такое отношение булева матрица не содержит нулевых строк. Иначе (пр$_A R \subset A$) отношение R называется *частичным* или *частично определенным*. Рассмотренное выше отношение R является частичным, так как пр$_A R \neq A$ ($a_4 \notin$ пр$_A R$).

2.2.2 Операции над бинарными отношениями

Обратным отношением для некоторого отношения $R \subseteq A \times B$ является отношение R^{-1}, определенное на $B \times A$ и образованное такими парами $(b, a) \in B \times A$, для которых $(a, b) \in R$. Матрица, представляющая отношение R^{-1}, получается транспонированием матрицы, представляющей R, т. е. заменой строк столбцами и наоборот.

Например, рассмотренному выше отношению $R = \{(a_1, b_1), (a_1, b_3), (a_2, b_1), (a_2, b_3), (a_2, b_4), (a_3 b_1), (a_3, b_4), (a_5, b_4)\}$ будет соответствовать обратное отношение $R^{-1} = \{(b_1, a_1), (b_3, a_1), (b_1, a_2), (b_3, a_2), (b_4, a_1), (b_1, a_3), (b_4, a_3), (b_4, a_5)\}$, представляемое матрицей

$$\begin{array}{cccccc}
a_1 & a_2 & a_3 & a_4 & a_5 & \\
1 & 1 & 1 & 0 & 0 & b_1 \\
0 & 0 & 0 & 0 & 0 & b_2 \\
1 & 1 & 0 & 0 & 0 & b_3 \\
0 & 1 & 1 & 0 & 1 & b_4
\end{array}$$

В графическом представлении отношения, обратного для $R \subseteq A \times B$, доли A и B меняются местами и соответственно дуги меняют ориентацию на противоположную (рисунок 2.3).

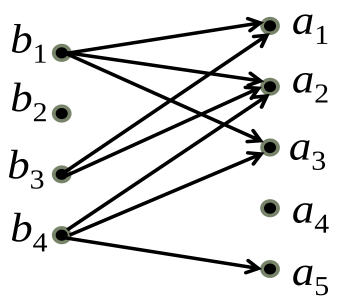


Рисунок 2.3 – Графическое представление отношения $R^{-1} \subseteq B \times A$

Поскольку всякое отношение есть некоторое множество пар, к отношениям применимы все теоретико-множественные операции, определенные для множеств, т. е. объединение, пересечение, разность, сумма и дополнение. При этом отношения, связываемые теоретико-множественными операциями, должны быть определены на одном и том же универсуме. Например, пусть отношения R_1 и R_2 определены на $A \times B$, тогда

$R_1 \cup R_2 = \{(a_i, b_j) /(a_i, b_j) \in R_1$ или $(a_i, b_j) \in R_2\}$;

$R_1 \cap R_2 = \{(a_i, b_j) /(a_i, b_j) \in R_1$ и $(a_i, b_j) \in R_2\}$;

$R_1 \setminus R_2 = \{(a_i, b_j) /(a_i, b_j) \in R_1$ и $(a_i, b_j) \notin R_2\}$;

$R_1 + R_2 = \{(a_i, b_j) /((a_i, b_j) \in R_1$ и $(a_i, b_j) \notin R_2)$ или $((a_i, b_j) \in R_2$ и $(a_i, b_j) \notin R_1)\}$;

$\overline{R} = \{(a_i, b_j) /(a_i, b_j) \notin R\}$ или $\overline{R} = A \times B \setminus R$.

Например, пусть отношения R_1 и R_2 заданы в матричном виде:

$$\boldsymbol{R_1} = \begin{array}{cccc} b_1 & b_2 & b_3 & \\ 1 & 1 & 0 & a_1 \\ 1 & 0 & 1 & a_2 \\ 1 & 1 & 0 & a_3 \\ 1 & 1 & 0 & a_4 \\ 0 & 0 & 1 & a_5 \end{array} \qquad \boldsymbol{R_2} = \begin{array}{cccc} b_1 & b_2 & b_3 & \\ 0 & 1 & 0 & a_1 \\ 1 & 0 & 1 & a_2 \\ 1 & 0 & 0 & a_3 \\ 0 & 0 & 1 & a_4 \\ 1 & 1 & 1 & a_5 \end{array}$$

тогда

$$\boldsymbol{R_1} \cap \boldsymbol{R_2} = \begin{array}{cccc} b_1 & b_2 & b_3 & \\ 0 & 1 & 0 & a_1 \\ 1 & 0 & 1 & a_2 \\ 1 & 0 & 0 & a_3 \\ 0 & 0 & 0 & a_4 \\ 0 & 0 & 1 & a_5 \end{array} \qquad \boldsymbol{R_2} + \boldsymbol{R_2} = \begin{array}{cccc} b_1 & b_2 & b_3 & \\ 1 & 0 & 0 & a_1 \\ 0 & 0 & 0 & a_2 \\ 0 & 1 & 0 & a_3 \\ 1 & 1 & 1 & a_4 \\ 1 & 1 & 0 & a_5 \end{array}$$

и

$$\begin{array}{cccc} b_1 & b_2 & b_3 & \\ 0 & 0 & 1 & a_1 \\ 0 & 1 & 0 & a_2 \\ 0 & 0 & 1 & a_3 \\ 0 & 0 & 1 & a_4 \\ 1 & 1 & 0 & a_5 \end{array} \qquad \boldsymbol{R}_2^{-1} = \begin{array}{cccccc} a_1 & a_2 & a_3 & a_4 & a_5 & \\ 0 & 1 & 1 & 0 & 1 & b_1 \\ 1 & 0 & 0 & 0 & 1 & b_2 \\ 0 & 1 & 0 & 1 & 1 & b_3 \end{array}$$

$$\overline{\boldsymbol{R}_1} = \ldots$$

Рассмотрим также специальную операцию, определенную только для отношений, – операцию *композиции*. Пусть заданы множества A, B, C и отношения $R \subseteq A \times B$ и $S \subseteq B \times C$. Композицией SR отношений S и R является такое отношение между элементами множеств A и C, что для всех $a \in A$ образ a относительно SR совпадает с образом подмножества $R(a) \subseteq B$ относительно S:

$$(SR)(a) \subseteq S(R(a)).$$

Другими словами, некоторая пара $(a, c) \in SR$, если существует $b \in B$, такое, что $(a, b) \in R$ и $(b, c) \in S$, или композицией SR является отношение, состоящее из пар $(a, S(R(a)))$. Процесс построения композиции SR отношений наглядно иллюстрируется графически (рисунок 2.4). Пара $(a, c) \in SR$, если вершины a и c окажутся связанными парой дуг через некоторую вершину b.

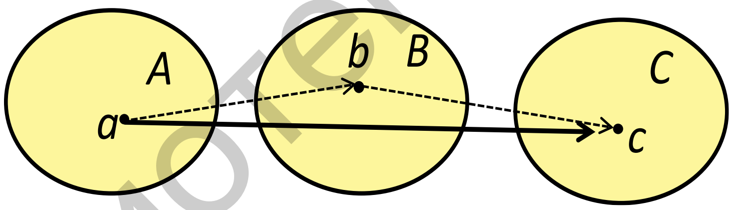


Рисунок 2.4 – Графическая иллюстрация получения композиции SR отношений $R \subseteq A \times B$ и $S \subseteq B \times C$

Важно отметить, что композиция SR отношений S и R существует, если R и S задают транзитивную связь между элементами множеств A, B и C: aRb и bSc. Очевидно, что для $R \subseteq A \times B$ и $S \subseteq B \times C$ существует композиция SR, но не существует композиция RS.

Матричное задание композиции RS получается логическим произведением матриц $\boldsymbol{R}$ и $\boldsymbol{S}$, которое находится путем логического умножения (конъюнкции) строк матрицы $\boldsymbol{R}$ на столбцы $\boldsymbol{S}$, при этом в качестве суммы используется логическая сумма (дизъюнкция). Следует обратить внимание также и на тот факт, что операция композиции определена для матриц согласованных размерностей: матрица $\boldsymbol{R}$ имеет размерность $n \times m$ (n столбцов и m строк), матрица $\boldsymbol{S}$ – $m \times k$. Результирующая матрица $\boldsymbol{S}\,\boldsymbol{R}$ будет иметь размерность $n \times k$.

Например, пусть отношения R и S заданы соответственно следующими матрицами:

$$\boldsymbol{R} = \begin{array}{cccc|c} b_1 & b_2 & b_3 & b_4 & \\ 1 & 0 & 1 & 0 & a_1 \\ 0 & 0 & 1 & 1 & a_2 \\ 1 & 1 & 0 & 0 & a_3 \\ 0 & 1 & 0 & 0 & a_4 \\ 0 & 0 & 0 & 1 & a_5 \end{array} \qquad \boldsymbol{S} = \begin{array}{ccc|c} c_1 & c_2 & c_3 & \\ 0 & 1 & 0 & b_1 \\ 1 & 1 & 0 & b_2 \\ 1 & 0 & 0 & b_3 \\ 0 & 0 & 1 & b_4 \end{array}$$

Композиция $S\,R$ этих отношений представится матрицей

$$\boldsymbol{S\,R} = \begin{array}{ccc|c} c_1 & c_2 & c_3 & \\ 1 & 1 & 0 & a_1 \\ 1 & 0 & 1 & a_2 \\ 1 & 1 & 0 & a_3 \\ 1 & 1 & 0 & a_4 \\ 0 & 0 & 1 & a_5 \end{array}$$

В этой матрице, например, первая строка получается следующим образом: $R(a_1) = \{b_1, b_3\}$ (см. первую строку матрицы $\boldsymbol{R}$), $S(\{b_1, b_3\}) = \{c_1, c_2\}$) (см. объединение строк b_1, b_3 матрицы $\boldsymbol{S}$). Или произведение первой строки матрицы $\boldsymbol{R}$ на первый столбец матрицы $\boldsymbol{S}$ определит элемент $s_1^1 \in \boldsymbol{S\,R}$, равный в данном случае 1, произведение первой строки матрицы $\boldsymbol{R}$ на второй столбец определит элемент $s_1^2 \in \boldsymbol{S\,R}$, равный в данном случае также 1. Получение композиции $S\,R$ заданных отношений, исходя из их графического задания приведено на рисунке 2.5.

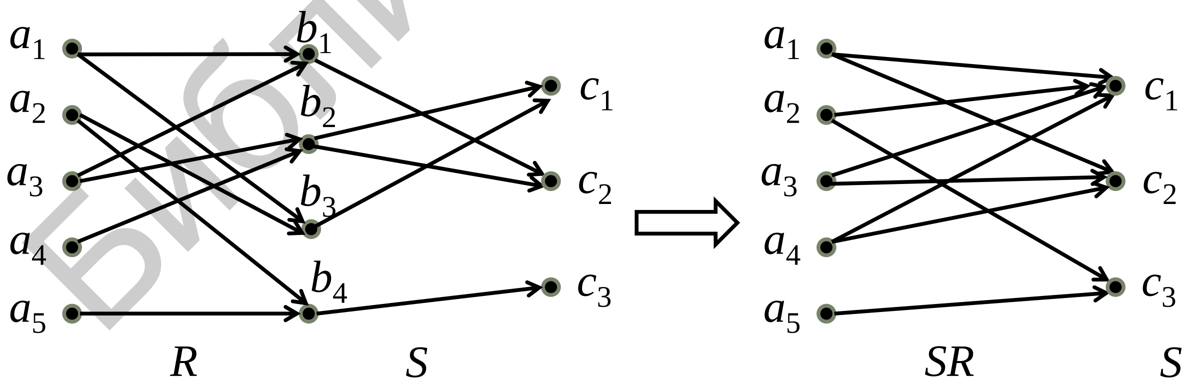


Рисунок 2.5 – Получение композиции $S\,R \subseteq A \times C$
отношений $R \subseteq A \times B$ и $S \subseteq B \times C$

Частным случаем является композиция отношений $R \subseteq A^2$ и $S \subseteq A^2$, заданных на множестве A, и, наконец, композиция отношения R с самим собой. Последняя композиция называется *степенью отношения* R – $R^2 = RR$ или $R^n = R^{n-1}R$ в общем случае – и имеет интересные приложения в теории графов (см. раздел 4).

Например, для отношения $R \subseteq A^2$, заданного следующей матрицей:

$$\boldsymbol{R} = \begin{array}{cccc|c} a_1 & a_2 & a_3 & a_4 & \\ 1 & 0 & 1 & 0 & a_1 \\ 0 & 0 & 1 & 1 & a_2 \\ 1 & 1 & 0 & 0 & a_3 \\ 0 & 1 & 0 & 0 & a_4 \end{array}$$

вторая степень имеет вид

$$\boldsymbol{R}^2 = \begin{array}{cccc|c} a_1 & a_2 & a_3 & a_4 & \\ 0 & 0 & 0 & 0 & a_1 \\ 1 & 0 & 0 & 0 & a_2 \\ 0 & 0 & 0 & 1 & a_3 \\ 0 & 0 & 0 & 0 & a_4 \end{array}$$

На графическом представлении отношения R дуги, которые задают пары из A^2, входящие в R^2, показаны штриховыми линиями (рисунок 2.6).

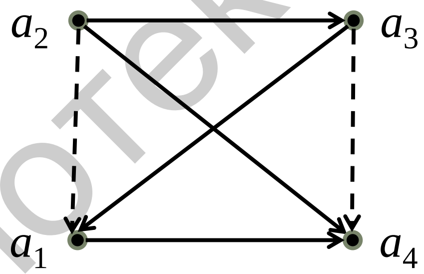


Рисунок 2.6 – Получение композиции R^2

2.2.3 Функциональные отношения и отображения

Отношение $R \subseteq A \times B$ называется *функциональным*, если образ каждого элемента $a \in A$ относительно отношения R содержит не более одного элемента ($|R(a)| = 1$). Это означает, что в функциональном отношении не существует пар $(a, b) \in R$ с одинаковой левой координатой и различными правыми координатами, т. е. если $(a, b) \in R$ и R является функциональным отношением, то в R не существует пары вида (a, c), где $b \neq c$. Каждая строка матрицы, представляющей функциональное отношение, имеет не более одной единицы. Примером может служить отношение R, заданное следующей матрицей:

$$\begin{array}{cccccc} & b_1 & b_2 & b_3 & \\ & 1 & 0 & 0 & a_1 \\ & 0 & 1 & 0 & a_2 \\ \boldsymbol{R} = & 1 & 0 & 0 & a_3 \\ & 0 & 0 & 1 & a_4 \\ & 0 & 1 & 0 & a_5 \end{array}$$

И из каждой вершины $a \in A$ графа, представляющего функциональное отношение, исходит не более одной дуги.

Если отношение R^{-1}, обратное функциональному отношению R, также является функциональным, то отношение R называется *взаимно однозначным*. Это условие выполняется, если каждый столбец матрицы $\boldsymbol{R}$ имеет не более одной единицы. Вышеприведенное отношение R не является *взаимно однозначным*.

Для всякого функционального отношения $R \subseteq A \times B$ можно определить *функцию*, связанную с этим отношением. Для обозначения функции используется запись $f: A \to B$. Если $(x, y) \in R$, то это можно выразить также в виде $y = f(x)$, где x является *аргументом*, а y – *значением функции* f.

Множества пр$_A R$ $(\{x \,/\, (x, y) \in R\})$ и пр$_B R$ $(\{y \,/\, (x, y) \in R\})$ задают и соответственно *области определения* и *значений* функции f. Если область определения функции f совпадает с множеством A, то функция является *всюду определенной*. Такая функция называется также *отображением* множества A в B. В противном случае функцию f называют *частичной*.

Если область значений функции f совпадает с множеством B, то функция называется отображением A на B или *сюръективным* отображением (*сюръекцией*).

Из определения *сюръективности* следует, что отображение $f: A \to B$ является сюръекцией, если для всех $b \in B$ имеет место $|f^{-1}(b)| \geq 1$, т. е. каждый элемент множества B имеет не менее одного прообраза. Каждый столбец матричного задания такого отображения A на B содержит не менее одной единицы. Обязательным условием существования отображения A на B является условие $|A| \geq |B|$. Оно вытекает из определений функциональности и сюръективности: $|R(a)| = 1$ для всех $a \in A$ и $|f^{-1}(b)| \geq 1$.

Из трех следующих отношений:

$$\boldsymbol{R} = \begin{array}{ccc|c} b_1 & b_2 & b_3 & \\ 1 & 0 & 0 & a_1 \\ 0 & 1 & 0 & a_2 \\ 0 & 0 & 1 & a_3 \\ 0 & 1 & 0 & a_4 \\ 0 & 0 & 0 & a_5 \end{array}$$

$$\boldsymbol{H} = \begin{array}{cccc|c} b_1 & b_2 & b_3 & b_4 & \\ 1 & 0 & 0 & 0 & a_1 \\ 0 & 1 & 0 & 0 & a_2 \\ 0 & 0 & 1 & 0 & a_3 \end{array} \qquad \boldsymbol{Q} = \begin{array}{ccc|c} b_1 & b_2 & b_3 & \\ 1 & 0 & 0 & a_1 \\ 0 & 0 & 1 & a_2 \\ 0 & 1 & 0 & a_3 \end{array}$$

отношения R и Q являются сюръективными, а H – нет.

Отображение $f : A \to B$ называется *инъективным* или *инъекцией*, если разные элементы из A отображаются в разные элементы из B или прообраз любого элемента $b \in B$ относительно такого отображения содержит не более одного элемента: $|f^{-1}(b)| \leq 1$. Соответственно каждый столбец матричного задания такого отображения f содержит не более одной единицы. Обязательным условием существования инъективного отображения A в B является $|A| \leq |B|$. В этом случае существует функция f^{-1}, которая является *обратной* к функции f. При этом если $y = f(x)$, то $x = f^{-1}(y)$, а мощность области определения функции f не должна превышать $|B|$.

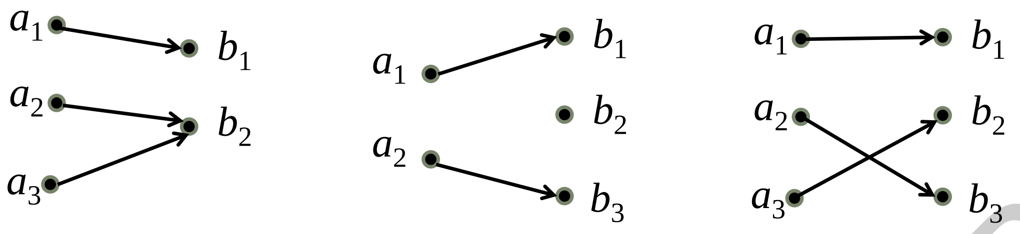
Из трех приведенных выше отношений только R не является инъективным.

Отображение $f: A \to B$ называется *биективным* или *биекцией*, если оно является сюръективным и инъективным отображением одновременно. При биективном отображении каждому элементу одного множества соответствует ровно один элемент другого множества. Биективное отображение обозначается как $f: A \leftrightarrow B$ и называется еще *взаимно однозначным отображением* или 1-1 *отображением*. Приведенное выше отображение Q является биективным.

Каждый столбец и каждая строка матричного задания биективного отображения содержат ровно по одной единице, число строк матрицы равно числу столбцов. Перестановкой строк и столбцов такую матрицу можно привести к диагональному виду.

На рисунке 2.7 приведена графическая иллюстрация рассмотренных видов отображений.

Если R – взаимно однозначное отношение между элементами одного и того же множества, т. е. $R \subseteq A \times A = A^2$, и, кроме того, R и R^{-1} всюду определены, то отображение, связанное с R, называется *перестановкой* или *подстановкой*.



а *б* *в*

а – сюръекция (но не инъекция); *б* – инъекция (но не сюръекция); *в* – биекция

Рисунок 2.7 – Элементы графического задания функциональных отображений

Рассмотрим примеры отображений $f : A \to B$.

1. Отображение $f : A \to A$ на множестве A действительных чисел. Функция $f(a) = a^2$ не сюръективна, так как отрицательные числа не входят в область значений функции, и не инъективна, так как обратное отношение не является функциональным.

2. Отображение $f : A \to B$, где A – множество всех, а B – положительных действительных чисел. Функция $f(a) = b$, где $b = |a|$, сюръективна, но не инъективна, так как обратное отношение не является функциональным.

3. Отображение $f : A \to B$, где A – множество символов некоторого алфавита, а B – множество натуральных чисел. Функция $f(a) = b$, где b – порядковый номер символа a, сюръективна, инъективна, а значит, и биективна.

4. Функция $f(a) = e^a$ сюръективна, инъективна, а значит, и биективна.

Отображение $f : A \to B$, где A и B – некоторые множества функций, называется *оператором*. Оператор преобразует одну функцию в другую. Примером оператора является оператор суперпозиции функций, где аргументами некоторых функций служат другие функции. Если имеется две функции $f : A \to B$ и $g : B \to C$, то суперпозиция функций обозначается через $f \circ g = g(f)$.

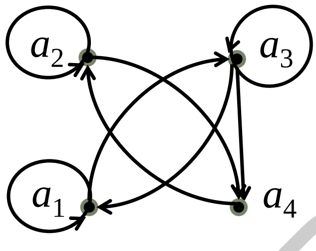
2.3 Бинарные отношения на множестве

Если $R \subseteq A \times A = A^2$, то R является бинарным отношением на множестве A. В матричном виде такое отношение представляется квадратной булевой матрицей $\boldsymbol{R}$. Элемент $r_{ij} \in \boldsymbol{R}$, стоящий на пересечении i-й строки и j-го столбца равен 1, если $a_i R a_j$, $a_i, a_j \in A$.

На рисунке 2.8 приведены матричное и графическое представления бинарного отношения $R = \{(a_1,a_1), (a_1,a_3), (a_2,a_2), (a_2,a_4), (a_3,a_1), (a_3,a_3), (a_3,a_4), (a_4,a_2)\}$ на множестве $A = \{a_1, a_2, a_3, a_4\}$.

$$\boldsymbol{R} = \begin{array}{cccc|c} a_1 & a_2 & a_3 & a_4 & \\ 1 & 0 & 1 & 0 & a_1 \\ 0 & 1 & 0 & 1 & a_2 \\ 1 & 0 & 1 & 1 & a_3 \\ 0 & 1 & 0 & 0 & a_4\,. \end{array}$$

а



б

а – матричное задание; *б* – графическое задание

Рисунок 2.8 – Пример бинарного отношения $R \subseteq A^2$

2.3.1 Свойства бинарных отношений на множестве

Определим некоторые свойства, которыми может обладать или не обладать бинарное отношение на множестве.

Рефлексивность (*Р*). Отношение $R \subseteq A^2$ называется рефлексивным, если для любого $a \in A$ имеет место $a\,R\,a$. Все элементы главной диагонали булевой матрицы $\boldsymbol{R}$ равны 1 ($r_{ij} = 1$, если $i = j$).

Иррефлексивность (*И*). Отношение $R \subseteq A^2$ называется иррефлексивным, если ни для одного $a \in A$ не имеет место $a\,R\,a$, т. е. если $a\,R\,b$, то $a \neq b$. Все элементы главной диагонали булевой матрицы $\boldsymbol{R}$ равны 0 ($r_{ij} = 0$, если $i = j$).

Например, отношения «≤» и «иметь общий делитель на множестве действительных чисел» рефлексивны; отношение «<» – иррефлексивно; отношение $R = \{(x, y)/\ x, y \in A$ и $y = x^2\}$ не обладает ни одним из этих свойств, а приведенное на рисунке 2.8 отношение R не является ни рефлексивным, ни иррефлексивным.

Симметричность (*С*). Отношение $R \subseteq A^2$ называется симметричным, если для любой пары $(a, b) \in A^2$ из $a\,R\,b$ следует $b\,R\,a$, т. е. если $a\,R\,b$, то и $b\,R\,a$. Булева матрица $\boldsymbol{R}$ симметрична относительно главной диагонали.

Из определения следует, что если отношение R обладает свойством симметричности, то $R = R^{-1}$.

Антисимметричность (*А*). Отношение $R \subseteq A^2$ называется антисимметричным, если для любой пары $(a, b) \in A^2$ из $a\,R\,b$ и $b\,R\,a$ следует, что $a = b$. В булевой матрице $\boldsymbol{R}$ не существует ни одной пары элементов, которые симметричны относительно главной диагонали и имеют значение 1.

Например, отношение «≤» антисимметрично; отношение параллельности на множестве прямых евклидовой плоскости симметрично, а приведенное на рисунке 2.8 отношение R не является ни симметричным, ни антисимметричным.

Транзитивность (*T*). Отношение $R \subseteq A^2$ называется транзитивным, если для любых $a,\ b,\ c \in A$ из $a\,R\,b$ и $b\,R\,c$ следует, что $a\,R\,c$.

Например, отношение «=» на множестве действительных чисел, отношение параллельности на множестве прямых евклидовой плоскости, отношение «быть подмножеством множества» рефлексивны, симметричны и транзитивны, а отношения «≤» на множестве действительных чисел и «быть сыном» на множестве «людей» иррефлексивны, антисимметричны и первое отношение транзитивно, второе – нет. Отношение делимости на множестве натуральных чисел рефлексивно и транзитивно. Приведенное на рисунке 2.8 отношение R также не является транзитивным.

Из приведенного выше определения следует, что условие $R^2 \subset R$ является необходимым и достаточным условием транзитивности отношения R. Легко установить по графическому (проверкой наличия замыканий путей длиной 2) или матричному заданию отношения (путем получения путей длиной 2 умножением матрицы $\boldsymbol{R}$ на саму себя).

Анализ графического представления отношения R (рисунок 2.8, *б*) позволяет установить, что для его транзитивного замыкания необходимо добавить в него пары (a_1, a_4), (a_3, a_2) и (a_4, a_4), а значит, отношение R не является транзитивным. В этом можно убедиться и по матричному заданию отношений R и R^2:

$$\boldsymbol{R} = \begin{array}{cccc|c} a_1 & a_2 & a_3 & a_4 & \\ 1 & 0 & 1 & 0 & a_1 \\ 0 & 1 & 0 & 1 & a_2 \\ 1 & 0 & 1 & 1 & a_3 \\ 0 & 1 & 0 & 0 & a_4 \end{array} \qquad \boldsymbol{R}^2 = \begin{array}{cccc|c} a_1 & a_2 & a_3 & a_4 & \\ 1 & 0 & 1 & \mathbf{1} & a_1 \\ 0 & 1 & 0 & 1 & a_2 \\ 1 & \mathbf{1} & 1 & 1 & a_3 \\ 0 & 1 & 0 & \mathbf{1} & a_4 \end{array}$$

Единицы, помеченные полужирным шрифтом в матрице $\boldsymbol{R}^2$, отсутствуют в матрице $\boldsymbol{R}$ и соответствуют замыканиям $a_1\,R\,a_3$ и $a_3\,R\,a_4$; $a_3\,R\,a_4$ и $a_4\,R\,a_2$; $a_4\,R\,a_2$ и $a_2\,R\,a_4$.

Транзитивное замыкание бинарного отношения R на множестве A есть наименьшее транзитивное отношение на множестве A, включающее R. Например, если A – множество населенных пунктов и $a_i\,R\,a_j$, если из пункта a_i существует автобусный маршрут в a_j, тогда транзитивным замыканием отношения R будет отношение R^t: $a_k\,R^t\,a_l$, если из пункта a_k возможно на автобусах добраться в пункт a_l.

Определение транзитивного замыкания сводится к последовательному поиску таких пар $(x, y) \in R$ и $(y, z) \in R$, что $(x, z) \notin R$, и добавлению в множество R замыкающей пары (x, z). Процедура выполняется до тех пор, пока такие пары находятся. Например, транзитивное замыкание отношения $R = \{(1, 2), (2, 3), (2, 4), (4, 5)\}$ на множестве $A = \{1, 2, 3, 4, 5\}$, дает отношение $R^t = \{(1, 2), (2, 3), (2, 4), (4, 5), \mathbf{(1, 3)}, \mathbf{(1, 4)}, \mathbf{(2, 5)}, \mathbf{(1, 5)}\}$. На графическом представлении отношения R^t (рисунок 2.9) пары $(x, y) \in A^2$, такие, что $(x, y) \notin R$, но $(x, y) \in R^t$, показаны пунктирной линией.

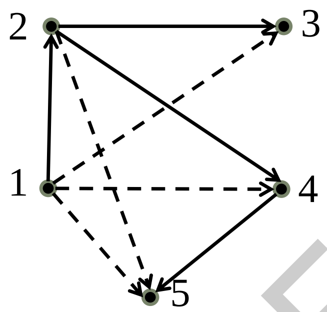


Рисунок 2.9 – Графическая иллюстрация примера транзитивного замыкания отношения

Дихотомия (*Д*). Отношение $R \subseteq A^2$ обладает свойством дихотомии, если для любых $a, b \in A$ имеет место (если $a \neq b$) либо $a R b$, либо $b R a$. Очевидно, что всякое отношение, обладающее свойством дихотомии, антисимметрично. Обратное в общем случае не верно.

Свойство дихотомии легко устанавливается просмотром пар элементов матричного представления отношения, симметричных относительно главной диагонали (они должны иметь разные значения), т. е. части матрицы выше и ниже главной диагонали должны быть взаимно инверсны.

Например, отношение «<» на множестве натуральных чисел обладает свойством дихотомии, а приведенное на рисунке 2.8 отношение R – нет (оно даже не является антисимметричным).

Ниже приведены некоторые важные для практических приложений типы бинарных отношений, характеризуемые определенным набором свойств.

2.3.2 Отношение эквивалентности

Рефлексивное, симметричное и транзитивное отношение $R \subseteq A^2$ (обладает свойствами Р+С+Т) называется *отношением эквивалентности*.

Примерами отношения эквивалентности являются равносильность формул, подобие геометрических фигур, принадлежность студентов к одной группе, обладание одинаковым цветом глаз, принадлежность населенных пунктов к одному району, чисел к одному классу вычетов и т. п.

Отношение эквивалентности на множестве делит его на непересекающиеся подмножества, называемые *классами эквивалентности*. Любые два элемента, принадлежащие одному и тому же классу эквивалентности, эквивалентны, а элементы, принадлежащие различным классам, не являются эквивалентными. Кроме того, каждый элемент $a \in A$ принадлежит только одному классу. Класс эквивалентности, содержащий элемент $a \in A$, обозначается через $[a]$ и представляет собой множество элементов $[a] = \{x / a\,R\,x,\ x \in A\}$. Элемент a называется представителем класса $[a]$.

Каждый класс эквивалентности в матричном виде представляется квадратной матрицей, все элементы которой равны 1.

Например, пусть на множестве $\mathbb{N}$ натуральных чисел задано отношение «$\equiv$» равенства чисел по модулю k: $n \equiv m \pmod{k}$, если $(m - n) = l\,k$. Это отношение является отношением эквивалентности на $\mathbb{N}$, если $l = 0, 1, 2, \ldots$. При $k = 2$ имеется два класса эквивалентности с представителями 0, 1 и 2:

$[0] = \{2n / n \in \mathbb{N}\}$ – множество четных чисел;

$[1] = \{2n + 1 / n \in \mathbb{N}\}$ – множество нечетных чисел.

Матричное представление этого отношения для случая $A = \{1, 2, 3, 4, 5, 6, 7, 8\}$ имеет вид

$$\boldsymbol{R} = \begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 2 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 3 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 4 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 5 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 6 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 7 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 8 \end{array}$$

По матричному представлению рассматриваемого отношения R легко заметить, что оно рефлексивно (все элементы по главной диагонали равны 1), симметрично (матрица симметрична относительно главной диагонали), транзитивно ($R^2 \subset R$).

Множество всех классов эквивалентности образует *фактор-множество* множества A по R и обозначается через A / R. Для приведенного примера отношения фактор-множество $A / R = \{\{1, 3, 5, 7\}, \{2, 4, 6, 8\}\}$.

Построим разбиение множества A по отношению эквивалентности R на этом множестве A. Для некоторого элемента $a \in A$ обозначим через $K(A)$ подмножество из A, состоящее из элементов $a_i \in A$, эквивалентных a ($a \sim a_i$): $K(a) =$
$= \{a_i \,/\, a \sim a_i,\ a_i \in A\}$.

Так как отношение эквивалентности транзитивно, то все пары элементов $a_i, a_j \in K(a)$ эквивалентны. Пусть некоторые a и a' не эквивалентны, покажем, что $K(a) \cap K(a') = \varnothing$, т. е. $K(a_1)$, $K(a_2)$, … задают разбиение множества A. Допустим противное, что $K(a) \cap K(a') \neq \varnothing$, тогда найдется $a'' \in K(a) \cap K(a')$, из последнего следует, что $a'' \sim a$ и $a'' \sim a'$. Отсюда, используя свойства симметричности и транзитивности, получаем противоречие: $a \sim a'$.

Ясно, что справедливо и обратное: каждому разбиению множества A соответствует некоторое отношение эквивалентности, определенное на этом множестве.

Для вышеприведенного примера отношения эквивалентности фактор-множество $A/R = \{\{1, 3, 5, 7\}, \{2, 4, 6, 8\}\}$. Если строки и столбцы матричного задания отношения эквивалентности переупорядочить в соответствии с разбиением на классы эквивалентности, то элементам каждого из классов эквивалентности будет соответствовать подматрица, все элементы которой равны 1. Например, для приведенного примера элементы этих подматриц выделены полужирным шрифтом:

$$\boldsymbol{R} = \begin{array}{cccccccc|c}
1 & 3 & 5 & 7 & 2 & 4 & 6 & 8 & \\
\mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 & 0 & 0 & 1 \\
\mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 & 0 & 0 & 3 \\
\mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 & 0 & 0 & 5 \\
\mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 & 0 & 0 & 7 \\
0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 2 \\
0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 4 \\
0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 6 \\
0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 8
\end{array}$$

Рефлексивное и симметричное отношение $R \subseteq A^2$ (обладает свойствами Р+С), не обладающее свойством транзитивности, называется *отношением толерантности*. Примером такого отношения является отношение *совместимости*: оно рефлексивно и симметрично, но в общем случае не транзитивно. Примерами отношения совместимости являются близость чисел, обладание общим делителем, знакомство людей и т. п.

2.3.3 Отношения порядка

Отношением порядка называется любое отношение $R \subseteq A^2$, обладающее свойствами антисимметричности и транзитивности (свойства А+Т). Рефлексивное отношение порядка (обладает свойствами Р+А+Т) называется отношением *нестрогого порядка*. Например, отношения «≤» (меньше или равно) и «≥» (больше или равно) для действительных чисел, так же как «⊆» и «⊇» для множеств, являются отношениями нестрогого порядка.

Иррефлексивное отношение порядка (обладает свойствами И+А+Т) называется отношением *строгого порядка*. Например, отношениями строгого порядка являются отношения $\subset$ и $\supset$ для множеств.

Отношение *полного порядка* обладает свойствами иррефлексивности, транзитивности и дихотомии (обладает свойствами И+А+Т+Д). Полный порядок называют еще *линейным* или *совершенным*. Отношение *частичного порядка* (строгого или нестрогого) обладает свойством антисимметричности, но не обладает свойством дихотомии.

Для множества действительных чисел отношения $>$ и $<$ являются отношениями полного порядка. Для булеана 2^A (семейства подмножеств некоторого множества A) отношение «⊆» является отношением частичного порядка. Например, $\{a_1, a_3\} \subseteq \{a_1, a_2, a_3\}$, а подмножества $\{a_1, a_3\}$ и $\{a_1, a_2, a_4\}$ не сравнимы.

Множество A, на котором задано отношение частичного порядка R (строгого или нестрогого), называется частично упорядоченным, и множество A является *полностью* упорядоченным, если на нем задано отношение полного порядка. В полностью упорядоченном множестве A любые два его элемента a и b находятся в отношении R, т. е. $a\,R\,b$ или $b\,R\,a$. При этом говорят, что a и b *сравнимы*. Если A содержит хотя бы одну пару элементов c и d, для которых не имеет место ни $c\,R\,d$, ни $d\,R\,c$, то множество M является частично упорядоченным, а указанные элементы c и d *не сравнимы*.

Необходимо помнить, что порядок элементов не является атрибутом самого множества. Одно и то же множество может быть упорядочено разными отношениями порядка. Обозначим через «$\leq_R$» отношение порядка. Пара $(A, \leq_R)$ задает множество A, упорядоченное отношением «$\leq_R$».

Говорят, что элемент a из $(A, \leq_R)$ предшествует элементу $b \in A$ (b следует за a), если $a \leq_R b$.

Элемент $a_0 \in A$ называется *максимальным элементом* упорядоченного множества $(A, \leq_R)$, если не существует элемента $a \in A$, такого, что $a_0 \leq_R a$ и $a_0 \neq a$. Если не существует $a \in A$, такого, что $a \leq_R a_0$ и $a_0 \neq a$, то a_0 называется *минимальным элементом* упорядоченного множества A. Максимального и/или минимального элементов для упорядоченного множества может и не существовать, или их может быть больше, чем один.

Например, отношения порядка, графическое представление которых приведено на рисунках 2.10, *а* и 2.10, *в*, имеют по одному максимальному и минимальному элементу: 1 и 0, (1,1) и (0,0), а отношение на рисунке 2.10, *б* имеет по два максимальных и минимальных элемента: a, b и e, f.

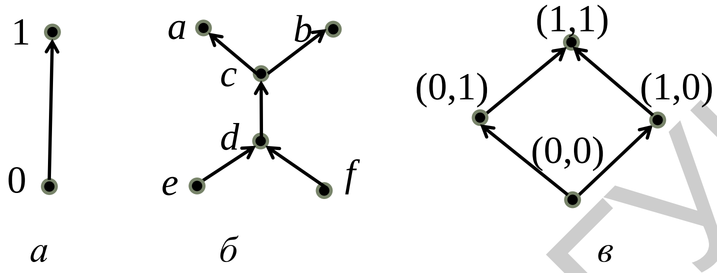


Рисунок 2.10 – Примеры отношений порядка

Порядок букв в алфавите и естественный порядок цифр являются полными порядками. На основе порядка, определенного на множестве букв, строится *лексикографический* порядок слов, используемый в словарях и определяемый следующим образом.

Обозначим это отношение порядка символом «$\prec$». Пусть имеются слова $w_1 = a_1a_2 \ldots a_m$ и $w_2 = b_1b_2 \ldots b_n$, где a_i, b_i – некоторые буквы алфавита. Тогда $w_1 \prec w_2$, если и только если:

1) либо $w_1 = pa_iq$, $w_2 = pb_ir$ и $a_i \prec b_i$, где p, q и r – некоторые слова, возможно, пустые, а a_i и b_i – буквы;

2) либо $w_2 = w_1p$, где p – непустое слово.

Например, «последний $\prec$ последовательность» и «буква $\prec$ букварь». В первом случае p = «послед», a_i = «н», b_i = «о», и в алфавите буква «н» перед буквой «о». Потому в словаре слово «последовательность» следует искать после слова «последний». Во втором случае w_1 = «буква» и p = «рь». Согласно лексикографическому порядку слово «букварь» будет следовать в словаре за словом «буква».

Аналогично можно лексикографически упорядочить числа в позиционных системах счисления (в двоичной, десятичной), как слова в алфавите цифр. Это упорядочение совпадет с упорядочением по отношению «$\leq$», если число разрядов у чисел будет равным (в ЭВМ это делается путем выравнивания).

Задания

1. Выполняются ли законы коммутативности, ассоциативности, дистрибутивности (относительно операций «$\cup$» и «$\cap$») для следующих декартовых произведений (доказать)?

$A \times B$;

$A \times (B \times C)$;

$A \times (B \cup C)$;

$A \times (B \cap C)$;

$(A \cup B) \times C$;

$(A \cap B) \times C$;

$(A \setminus B) \times C$.

2. Пусть: $A = \{a, b, c\}$, $B = \{1, 2, 3, 4\}$, $C = \{p, q, s\}$;

$R \subseteq A \times B$ и $R = \{(a,1), (b,3), (b,4), (c,1), (c,4)\}$;

$S \subseteq B \times C$ и $S = \{(1, p), (1, q), (2, p), (3, p), (4, q)\}$.

2.1. Найти:

– проекции: $Пр_A\,(b, 4)$, $Пр_B\,(b, 4)$, $Пр_A\{(b, 4), (c,1)\}$, $Пр_B\{(b, 4), (3,1)\}$;

– образы: $R(b)$, $R(c)$, $S(4)$, $R(\{a, b\})$, $R(\{b, c\})$, $S(\{1,4\})$;

– прообразы: $R^{-1}(2)$, $R^{-1}(1)$, $S^{-1}(q)$, $R^{-1}(\{3, 4\})$, $R^{-1}(\{1, 3, 4\})$.

2.2. Построить матрицы образов отношений.

2.3. Задать отношения в графическом и матричном видах.

2.4. Вычислить S^{-1}, $R^{-1}\,S\,R$.

2.5. Проверить, что $(S\,R)^{-1} = R^{-1}\,S^{-1}$.

2.6. Найти области определения и значений для отношений R и S.

3. Для отношений $R \subseteq A \times B$ и $Q \subseteq A \times B$, заданных в матричном виде

R

b_1	b_2	b_3	
1	0	1	a_1
0	1	0	a_2
1	0	0	a_3
0	0	0	a_4

Q

b_1	b_2	b_3	
0	0	1	a_1
1	1	0	a_2
0	1	0	a_3
0	1	1	a_4

Найти:

– объединение и пересечение;

– симметрическую разность;

– дополнение отношений R и Q;

– отношения, обратные R и Q.

4. Доказать следующие тождества:

$A \times B = (A \times D) \cap (C \times B)$ для $A \subseteq C$ и $B \subseteq D$;

$(A \times B) \cup (C \times D) \subseteq (A \cup C) \times (B \cup D)$.

5. Доказать следующие тождества, верные для произвольных отношений $R \subseteq A \times B$, $Q_1 \subseteq B \times C$, $Q_2 \subseteq B \times C$, $S \subseteq C \times D$:

$R(Q_1 \cup Q_2) = RQ_1 \cup RQ_2$;

$R(Q_1 Q_2) = (RQ_1)Q_2$;

$R(Q_1 \cap Q_2) = RQ_1 \cap RQ_2$;

$(Q_1 \cup Q_2)S = Q_1S \cup Q_2S$;

$(Q_1 \cap Q_2)S = Q_1S \cap Q_2S$;

$(R^{-1})^{-1} = R$;

$(R_1 \cup R_2)^{-1} = R_1^{-1} \cup R_2^{-1}$;

$(R_1 \cap R_2)^{-1} = R_1^{-1} \cap R_2^{-1}$;

$\neg(R^{-1}) = (\neg R)^{-1}$.

6. Сколько разных отношений можно построить на декартовом произведении $A \times B$?

7. Найти композицию отношений $R \subseteq A \times B$ и $Q \subseteq B \times C$, заданных в матричном виде:

$$R = \begin{array}{ccccc} b_1 & b_2 & b_3 & b_4 & \\ 1 & 0 & 1 & 0 & a_1 \\ 0 & 1 & 0 & 0 & a_2 \\ 1 & 0 & 0 & 1 & a_3 \\ 0 & 0 & 0 & 0 & a_4 \\ 0 & 0 & 0 & 1 & a_5 \end{array} \qquad Q = \begin{array}{cccc} c_1 & c_2 & c_3 & \\ 0 & 1 & 0 & b_1 \\ 1 & 1 & 0 & b_2 \\ 0 & 0 & 1 & b_3 \\ 0 & 0 & 1 & b_4 \end{array}$$

8. Являются ли следующие отношения функциональными (почему)?

$$\begin{array}{ccccc} y_1 & y_2 & y_3 & y_4 & \\ 0 & 0 & 0 & 1 & x_1 \\ 0 & 0 & 0 & 0 & x_2 \\ 1 & 0 & 0 & 0 & x_3 \\ 0 & 1 & 0 & 0 & x_4 \end{array} \qquad \begin{array}{ccccc} y_1 & y_2 & y_3 & y_4 & \\ 1 & 0 & 0 & 1 & x_1 \\ 0 & 0 & 1 & 0 & x_2 \\ 0 & 1 & 0 & 0 & x_3 \\ 0 & 0 & 1 & 0 & x_4 \end{array}$$

9. Для отношений $R_1 = \{(a, a), (a, b), (b, d)\}$ и $R_2 = \{(a, a), (a, b), (b, d)\}$ на $A = \{a, b, c, d\}$ найти (в матричной и графической формах):

$R_1 R_2$;

$R_2 R_1$;

$R_2^2 = R_2 R_2$;

$R_1 \cup R_2$;

R_1^{-1}.

10. Для отношений $R \subseteq A \times B$ и $G \subseteq A \times B$ ($A = \{a, b, c\}$, $B = \{1, 2, 3\}$):

$$\boldsymbol{R} = \begin{array}{ccc|c} 1 & 2 & 3 & \\ \hline 0 & 1 & 0 & a \\ 1 & 1 & 0 & b \\ 0 & 0 & 0 & c \end{array} \qquad \boldsymbol{G} = \begin{array}{ccc|c} 1 & 2 & 3 & \\ \hline 0 & 1 & 1 & a \\ 1 & 1 & 0 & b \\ 1 & 0 & 1 & c \end{array} \qquad \boldsymbol{H} = \begin{array}{cc|c} a & b & \\ \hline 1 & 1 & 1 \\ 0 & 0 & 2 \\ 1 & 0 & 3 \end{array}$$

10.1. Найти симметрическую разность отношений R и G.

10.2. Найти области определения и значений отношений R, G, H, R^{-1}.

10.3. Определить, какие из отношений R, G и H являются функциональными.

10.4. Найти композицию HR в матричном и графическом виде.

11. Пусть $R \subseteq X \times Y$ – отношение xRy между элементами множеств X и Y. В каком из следующих случаев это отношение можно рассматривать как функцию $y = R(x)$:

– X – множество студентов, Y – учебных дисциплин, xRy означает, что «x изучает y»;

– X – множество студентов вуза, Y – учебных групп, xRy означает, что «x – студент группы y».

12. Найти композицию R^2 в матричном и графическом виде:

$$\boldsymbol{R} = \begin{array}{ccccc|c} a & b & c & d & e & \\ \hline 0 & 1 & 0 & 0 & 1 & a \\ 1 & 0 & 0 & 0 & 0 & b \\ 0 & 0 & 0 & 1 & 0 & c \\ 0 & 1 & 0 & 0 & 1 & d \\ 0 & 0 & 1 & 0 & 0 & e \end{array}$$

13. Пусть $R \subseteq \{1, 2, 3\} \times \{1, 2, 3, 4\}$, $R = \{(1,1), (2,3), (2,4), (3,1), (3,4)\}$; $S \subseteq \{1, 2, 3, 4\} \times \{1, 2\}$, $S = \{(1,1), (1,2), (2,1), (3,1), (4,2)\}$. Вычислить $S\,R$, R^{-1} и S^{-1}. Проверить равенство $(S\,R)^{-1} = R^{-1}\,S^{-1}$.

14. Пусть $X = \{x_1, x_2, x_3, x_4\}$, $Y = \{y_1, y_2, y_3\}$. Задать некоторое частично определенное функциональное отношение $R \subset X \times Y$. Проверить, является ли оно биективным и/или сюръективным. Найти отношение, обратное R.

15. Пусть $f : A \to B$ и $g: B \to C$ – функции. Доказать, что

– если f и g инъективны, то $g\,f$ тоже инъективна;

– если f и g сюръективны, то $g\,f$ тоже сюръективна;

– если f и g обратимые функции, то $(g\,f)^{-1} = f^{-1}\,g^{-1}$.

16. Являются ли следующие функции сюръективными, инъективными, биективными:

$f(a) = a^2$ на множестве A действительных чисел;

$f(a) = b$ $(b = |a|)$, на $A \times B$, если A – множество действительных чисел, а B – множество положительных действительных чисел.

17. На множестве натуральных чисел $\mathbb{N}$ определено отношение R «быть равным по модулю k». Задать отношения в матричном виде и показать, что R – отношение эквивалентности и найти классы эквивалентности. Если

$k = 2$ и $N = \{1, 2, 3,\ldots, 10\}$;

$k = 3$ и $N = \{1, 2, 3,\ldots, 12\}$.

18. Определить, является ли отношение $R \subset A^2$

$$\boldsymbol{R} = \begin{array}{cccccc|c} a & b & c & d & e & f & \\ 1 & 0 & 1 & 1 & 0 & 1 & a \\ 0 & 1 & 0 & 0 & 1 & 1 & b \\ 0 & 0 & 1 & 0 & 0 & 1 & c \\ 0 & 0 & 1 & 1 & 0 & 0 & d \\ 0 & 0 & 0 & 0 & 1 & 0 & e \\ 0 & 0 & 0 & 0 & 1 & 1 & f \end{array}$$

отношением порядка. Если да, то сказать какого и упорядочить элементы из A этим отношением.

19. Какими свойствами (Р, И, С, А, Т, Д) обладают отношения:

– «x делит y» на множестве $\mathbb{N}$ натуральных чисел;

– «$x \neq y$» на множестве $\mathbb{Z}$ целых чисел;

– следующие отношения на множестве A:

$$\begin{array}{cccc} a_1 & a_2 & a_3 & a_4 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array} \qquad \begin{array}{cccc} a_1 & a_2 & a_3 & a_4 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array} \qquad \begin{array}{cccc} a_1 & a_2 & a_3 & a_4 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{array}$$

$$
\boldsymbol{R} = \begin{array}{ccccc|c}
a_1 & a_2 & a_3 & a_4 & a_5 & \\
1 & 0 & 1 & 0 & 0 & a_1 \\
0 & 0 & 1 & 1 & 0 & a_2 \\
1 & 1 & 0 & 0 & 0 & a_3 \\
0 & 1 & 0 & 0 & 1 & a_4 \\
0 & 0 & 0 & 1 & 0 & a_5
\end{array}
\qquad
\boldsymbol{S} = \begin{array}{ccccc|c}
a_1 & a_2 & a_3 & a_4 & a_5 & \\
1 & 0 & 0 & 1 & 1 & a_1 \\
1 & 1 & 0 & 0 & 1 & a_2 \\
1 & 1 & 1 & 1 & 0 & a_3 \\
0 & 1 & 0 & 1 & 0 & a_4 \\
0 & 0 & 1 & 1 & 1 & a_5
\end{array}
$$

20. Построить рефлексивное, симметричное и транзитивное замыкания вышеприведенных отношений.

21. Проверить, является ли отношение «делить нацело» на множестве $N = \{1, 2, 3,\ldots, 12\}$ отношением порядка, и если да, то задать его в графическом виде, найти максимальный и минимальный элементы.

22. Являются ли следующие отношения R и S на $A = \{a, b, c, d, e\}$ отношениями эквивалентности. Если да, то найти классы эквивалентности:

$R =\{(a, a), (a, c), (b, b), (b, e), (c, a), (c, c), (d, d), (e, b), (e, e)\}$;

$S =\{(a, a), (a, c), (b, b), (b, e), (c, a), (c, c), (c, d), (d, c), (d, d), (e, b), (e, e)\}$.

23. Задать в графическом виде (и указать минимальный и максимальный элементы) следующие частично упорядоченные множества:

– множество {1, 2, 3, 5, 6, 10, 15, 30} с отношением «x делит y»;

– булеан на {1, 2, 3} с отношением «X – подмножество Y»;

– множество A= {= {a, b, c, d, e, f , g, h} с отношением «$\subseteq$».

24. Является ли следующее отношение отношением эквивалентности. Если да, то найти классы эквивалентности:

$$
\boldsymbol{R} = \begin{array}{ccccccc|c}
a & b & c & d & e & f & g & \\
1 & 0 & 1 & 1 & 0 & 0 & 0 & a \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & b \\
1 & 0 & 1 & 1 & 0 & 0 & 0 & c \\
1 & 0 & 1 & 1 & 0 & 0 & 0 & d \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & e \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & f \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & g
\end{array}
$$

25. Построить бинарные отношения, обладающие свойствами:

– рефлексивное, симметричное, не транзитивное;

– не рефлексивное, антисимметричное, не транзитивное;

– рефлексивное, не симметричное, транзитивное.

26. К каким типам отношений (эквивалентности; порядка: строгого, нестрогого) относятся следующие отношения:

– отношение равносильности на множестве формул;

– отношение «<» на множестве векторов длиной n, компонентами которых являются натуральные числа, «<» определяется следующим образом: $(a_1, \ldots, a_n) < (b_1, \ldots, b_n)$, если $(a_1, \ldots, a_n) \leq (b_1, \ldots, b_n)$ и хотя бы для одной координаты i выполняется $a_i < b_i$;

– отношение предшествования на множестве слов упорядоченного конечного алфавита.

3 КОМБИНАТОРНЫЕ ЗАДАЧИ И ВЫЧИСЛИТЕЛЬНАЯ СЛОЖНОСТЬ

В дискретной математике выделяется класс комбинаторно-логических задач, решение которых связано с перебором комбинаций дискретных объектов и анализом возникающих вариантов. Количество исследуемых комбинаций может быть очень большим, и их перебор далеко не всегда удается сократить, поэтому комбинаторные задачи в общем случае относятся в математике к числу труднейших.

Отличительной чертой комбинаторных задач является высокая трудоемкость их решения. В отличие от задач, где решение получается с помощью целенаправленной вычислительной процедуры, однозначно ведущей к цели, решение комбинаторных задач часто сводится к полному перебору конструкций некоторого типа, среди которых находится решение рассматриваемой задачи. Процесс поиска решения считается завершенным, как только выясняется, что найденная конструкция является решением.

Далее рассматривается два типа комбинаторных задач: задачи перечислительной комбинаторики и оптимизационные комбинаторные задачи. В последние годы решению комбинаторных задач уделяется все большее внимание в связи с их многочисленными приложениями в дискретной математике: для решения транспортных задач, задач теории расписаний, планов производства, теории информации и т. д.

3.1 Перечислительная комбинаторика

Согласно американскому словарю комбинаторика определяется как «раздел математики, изучающий составление, перечисление и свойства разбиений, вариаций, сочетаний и перестановок из конечного числа элементов при различных условиях».

Возникновение комбинаторики как науки относят к XVI в., когда в жизни европейского общества большое место занимали азартные игры (карты, кости), всевозможные лотереи. Первоначально комбинаторные задачи касались в основном азартных игр – вопросов, сколькими способами можно набрать заданное число очков, бросая 2, 3 кости, может выпасть два короля в карточной игре и т. д. Эти азартные игры и были движущей силой развития комбинаторики и теории вероятностей.

Одним из первых подсчетом числа возможных комбинаций при игре в кости занялся итальянский математик Тарталья, который составил таблицу, показывающую, сколькими способами может выпасть r костей. Теоретическими исследованиями комбинаторики занимались французские математики Блез Паскаль и Пьер Ферма, дальнейшие исследования связаны с именами Д. Бернулли, Г. В. Лейбница, Леонарда Эйлера.

3.1.1 Основные правила и конфигурации

Перечислительная комбинаторика рассматривает задачи о перечислении или подсчете числа различных конфигураций, состоящих из элементов конечных множеств, на которые могут накладываться некоторые ограничения. Большинство простейших задач комбинаторики, сводящихся к подсчету числа конфигураций, решается с помощью двух основных правил – правила суммы и правила произведения.

Правило суммы. Если некоторый объект A можно выбрать n способами, а другой объект B можно выбрать m способами, то выбор «либо A, либо B» можно осуществить $n + m$ способами. При этом существенно, что выборы объектов A и B представляют собой независимые события.

Например, если лекции по матанализу посещают 25 студентов-программистов, а лекции по бухучету – 40 студентов-экономистов, то сколько студентов могут посетить эти лекции, если они проходят одновременно? Ответ: 25 + 40 = 65 студентов.

Правило произведения. Если некоторый объект A можно выбрать n способами и после каждого выбора объекта A объект B можно выбрать m способами, то выбор пары «B после A» можно осуществить $n\,m$ способами.

Например, если имеется 4 различных конверта и 5 марок, то сколькими способами можно выбрать конверт и марку для отсылки письма? Ответ: $4 \times 5 =$ $= 20$ способов.

*Теоретико-множественная интерпретация правила суммы***.** Если рассматривать A^t и B^t как множества исходов выбора соответственно объектов A и B, $|A^t| = n$, $|B^t| = m$, то, поскольку события выбора A и B не связаны друг с другом, можно считать, что соответствующие множества не пересекаются. Тогда $|A^t \cup B^t| = |A^t| + |B^t|$, т. е. множество $A^t \cup B^t$ содержит $n + m$ элементов. Это означает, что существует $n + m$ возможных исходов выбора «A или B».

Теоретико-множественную интерпретацию правила произведения также можно сформулировать на языке теории множеств. Пусть A_i^t обозначает множество n_i исходов i-го события. Тогда любую последовательность n событий можно рассматривать как элемент декартова произведения $A_1^t \times A_2^t \times \ldots \times A_n^t$, мощность которого равна $|A_1^t||A_2^t| \ldots |A_n^t|$.

Для формулировки и решения комбинаторных задач используют различные модели комбинаторных конфигураций. При решении задач на подсчет количества способов необходимо четко указывать тип уточнения формулировки. Чтобы различать на терминологическом уровне тип конкретной задачи, введем несколько определений. Начнем с вспомогательных терминов. Предположим, что мы берем m объектов $x_1, x_2, \ldots x_m$ из множества X мощностью n. Каждый такой набор объектов принято называть *выборкой* объема m из n объектов или (n, m)-выборкой.

Выборка называется *упорядоченной,* если порядок следования объектов в ней существенен. При этом две упорядоченные выборки, различающиеся лишь порядком следования объектов, считаются разными.

Если же порядок следования объектов в выборке не имеет значения, то выборка называется *неупорядоченной.*

Размещением называется упорядоченная выборка объектов из некоторого множества. *Сочетанием* называется неупорядоченная выборка объектов из некоторого множества.

В размещениях и сочетаниях могут допускаться и не допускаться повторения некоторых объектов.

(n, m)-*размещением с повторениями* называется упорядоченная (n, m)-выборка, объекты в которой могут повторяться.

(n, m)-*размещением без повторений* называется упорядоченная (n, m)-выборка, объекты в которой не повторяются.

(n, m)-*сочетанием с повторениями* называется неупорядоченная (n, m)-выборка с повторяющимися объектами.

(n, m)-*сочетанием без повторений* называется неупорядоченная (n, m)-выборка без повторяющихся объектов.

3.1.2 Подсчет числа конфигураций

Рассмотрим простейшие задачи подсчета числа конфигураций.

Число размещений $U(n, m)$ *с повторениями* показывает, сколькими способами можно выбрать m объектов из объектов n типов, причем объектов одного и того же типа может быть сколь угодно много. Очевидно, что первый из m объектов можно выбрать n способами, каждый из последующих также n способами. Следовательно, по правилу произведения имеем, что m объектов из объектов n типов можно выбрать следующим числом способов:

$$U(n, m) = n^m.$$

Пример 1. Сколькими способами можно разместить k предметов по l ящикам? Причем в одном ящике может быть сколь угодно много (до k) объектов. Очевидно, что для каждого из k предметов имеется l вариантов размещения, следовательно, по правилу произведения имеем $U(l, k) = l^k$.

Пример 2. Секретный замок открывается тогда, когда набран заданный код («тайное слово»). На диск нанесено 10 цифр (или букв), код состоит из 5 цифр. Общее число возможных комбинаций равно $U(10, 5) = 10^5$.

Число размещений без повторений $A(n, m)$ показывает, сколькими способами можно выбрать m объектов из n возможных (при этом считается, что $n \geq m$). Очевидно, что первый из m объектов можно выбрать n способами, второй – $(n – 1)$ способами, третий – $(n – 2)$ способами, соответственно m-й – $(n – (m – 1))$ способами. Следовательно, по правилу произведения имеем, что m объектов из n возможных можно выбрать следующим числом способов:

$$A(n, m) = n\,(n-1) \cdot \ldots \cdot (n-m+1) = \frac{n!}{(n-m)!}.$$

Пример 3. Сколькими способами можно разместить k предметов по l ящикам, не более чем по одному предмету в ящик (считается, что $l \geq k$)? Очевидно, что искомое число способов равно числу размещений k предметов:

$$A(l, k) = l\,(l-1) \cdot \ldots \cdot (l-k+1) = \frac{l!}{(l-k)!}.$$

Пример 4. Сколько существует различных четырехбуквенных последовательностей из неповторяющихся шести букв *а, б, г, о, и, м*? Искомое число последовательностей равно числу размещений без повторений четырех предметов:

$$A(6, 4) = \frac{6!}{(6-4)!} = 360.$$

При $m = n$ размещение без повторений называется *перестановкой*.

Числом перестановок $P(n)$ является число различных последовательностей длиной n, которые можно составить из n объектов. В последовательности всего n позиций. Зафиксируем один объект. Его можно разместить в одну из n позиций, т. е. имеется n вариантов размещения. Для следующего объекта имеется $n – 1$ вариантов размещения по незанятым позициям и т. д., и, наконец, для n -го объекта – один вариант. Таким образом, по правилу произведения имеем

$$P(n) = n\,(n-1) \cdot \ldots \cdot 2 \cdot 1 = n!$$

Различные перестановки отличаются только порядком выбора объектов, следовательно, число размещений без повторений $A(m, n)$ может быть выражено через числа перестановок следующим образом:

$$A(n, m) = \frac{n!}{(n-m)!} = \frac{P(n)}{P(n-m)}.$$

Пример 5. В научном обществе из 25 членов необходимо выбрать президента, вице-президента и ученого секретаря. Выбор этих трех человек может быть сделан следующим числом способов: $A(25, 3) = 25 \times 24 \times 23$.

Пример 6. Восемь ладей на шахматной доске можно разместить так, чтобы они не били друг друга, следующим числом способов (по одной ладье на каждой вертикали и горизонтали, т. е. по одной из диагоналей):

$$A(8, 8) = P(8) = 8! = 40\,320.$$

Перестановки с повторениями. Предположим, что из n имеющихся объектов n_1 имеет тип 1, n_2 имеет тип 2, … , n_k имеет тип k. Объекты одного типа неразличимы. Всего имеется $P(n) = n!$ перестановок n объектов, но не все они различны. Возьмем, к примеру, перестановку

$$\underbrace{a\,a \dots a}_{n_1}\ \underbrace{b\,b \dots b}_{n_2} \dots \underbrace{x\,x \dots x}_{n_k}.$$

Перестановки внутри отдельных групп не изменяют общего числа перестановок, а их можно произвести $n_1!\, n_2! \dots n_k!$ способами. Таким образом, множество $n!$ перестановок разделяется на группы таких одинаковых перестановок. Следовательно, число всех перестановок с повторениями равно

$$P(n_1, n_2, \dots, n_k) = \frac{n!}{n_1!\,n_2!\dots n_k!}.$$

Пример 7. Подсчитаем число перестановок букв, входящих в слово «Миссисипи» («и» повторяется 4 раза, «с» – 3, «м» и «п» – по одному разу):

$$P(4, 3, 1, 1) = \frac{9!}{4!3!} = 2520.$$

В тех случаях, когда не важен порядок выбора объектов и соответственно их положение в выборке, говорят о сочетаниях.

Число сочетаний $C(n, m)$ без повторений объектов (обозначается также C_n^m или $\binom{n}{m}$) показывает, сколькими способами из n объектов можно выбрать m объектов. Каждое из $C(n, m)$ сочетаний объединяет размещения без повторений, различающиеся только перестановками одних и тех же m объектов. Следовательно,

$$C(n,\ m) = \frac{A(n,m)}{P(m)} = \frac{n!}{(n-m)!\,m!}\ .$$

Пример 8. Генуэзская лотерея. В средние века была распространена лотерея, суть которой заключалась в следующем. Продавались билеты, на которых было от одного до пяти чисел от 1 до 90. В день розыгрыша вытягивались пять чисел, выигрывали те билеты, все числа на которых были среди пяти вытянутых. Например, выигрывал билет с числами 9, 40, 88, если вытянуты были, например, числа: 1, 9, 22, 40, 88. Если участник лотереи покупал билет с одним числом, то в случае выигрыша он получал 14 стоимостей билета, с двумя числами (амбо) – то 240, тремя (терн) – то 4800, четырьмя (катерн) – 75 000, пятью (квин) – 1 000 000 стоимостей билета. Найдем соотношение числа «счастливых» исходов к общему числу исходов. Число различных «пятерок» равно числу сочетаний

$$C(90,\ 5) = \frac{90!}{85!\,5!} = \frac{90\times 89\times 88\times 87\times 86}{2\times 3\times 4\times 5}.$$

Билет с одним числом фиксирует только это одно число, остальные четыре могут быть любыми из 89 оставшихся чисел. Соответственно число «выигрышных» комбинаций

$$C(89,\ 4) = \frac{89!}{85!\,4!} = \frac{89\times 88\times 87\times 86}{2\times 3\times 4}.$$

Соотношение числа благоприятных комбинаций к общему числу комбинаций в этом случае будет

$$\frac{C(89{,}4)}{C(90{,}5)} = \frac{5}{90} = 1/18.$$

Соотношение числа благоприятных комбинаций при игре Амбо к общему числу комбинаций будет

$$\frac{C(88{,}3)}{C(90{,}5)} = \frac{4\times 5}{90\times 89} = 2/801.$$

Совсем не выгодны терн и далее, их шансы выиграть равны 1/11 748, 1/511 038, 1/43 949 268.

Число сочетаний $F(n,\ m)$ с повторениями объектов показывает, сколькими способами из n типов объектов можно выбрать m объектов, среди которых может быть любое число экземпляров объектов одного и того же типа. Имеется n типов объектов: $C = \{c_1,\ c_2,\ \ldots,\ c_n\}$. Каждое сочетание состоит из m объектов $c_{i1},\ c_{i2},\ \ldots,\ c_{im}$, где $c_{ij} \in C$ (могут быть и одинаковые c_{ij}), поставим ему в соответствие набор $K = (k_1, k_2,\ldots, k_n)$ чисел, указывающих увеличенное на 1 число повторов каждого элемента $c_{ij} \in C$ в выбранном сочетании. При этом $k_1 + k_2 +\ldots+ k_n = n + m$. Например, пусть $C = \{a, b, c, d\}$, $m = 5$, $n = 4$, тогда сочетанию (a, c, c, d, d) сопоставляется набор $(2, 1, 3, 3)$, т. е. элементы a, b, c, d из C повторяются в выбранном сочетании соответственно 1, 0, 2, 2 раза. Набору K поставим в соответствие последовательность длины $n + m$ звездочек, разделенных вертикальными черточками на n непустых частей, состоящих соответственно из $k_1, k_2,\ldots, k_n$ звездочек. Например, для нашего примера получим

$$* * | * | * * * | * * *$$

Каждому разбиению числа $n + m$ на n ненулевых слагаемых взаимно однозначно соответствует распределение $n - 1$ разделителей, которые можно расставить в $n+m-1$ пробелах между звездочками C_{n+m-1}^{n-1} способами. Следовательно, число сочетаний с повторениями

$$F(n, m) = C_{n+m-1}^{n-1} = \frac{(n+m-1)!}{m!(n-1)!}.$$

Пример 9. В магазине продавалось четыре сорта пирожных, тогда, например, семь пирожныхможно купить следующим числом способов:

$$F(4, 7) = C(10, 7) = \frac{10!}{7!3!} = 120.$$

3.1.3 Связь числа сочетаний с биномиальными коэффициентами

Числа $C(n, m)$ представляют собой коэффициенты многочлена, получающегося при раскрытии скобок в биноме Ньютона $(a + b)^n$. Бином Ньютона – формула для разложения на отдельные слагаемые целой неотрицательной степени суммы двух переменных:

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k = \binom{n}{0} a^n + \binom{n}{1} a^{n-1} b + \ldots + \binom{n}{k} a^{n-k} b^k + \ldots + \binom{n}{n} b^n,$$

где $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ – *биномиальные коэффициенты*; n – положительное целое число.

Заметим, что сумма показателей степеней для a и b в каждом члене разложения постоянна и равна n.

Число членов, содержащих k множителей b и $n - k$ множителей a, равно числу способов выбора k мест из n возможных, т. е. C_n^k.

При $a = b = 1$ бином Ньютона дает

$$2^n = \sum_{k=0}^{n} C_n^k = C_n^0 + C_n^1 + \ldots + C_n^n .$$

Комбинаторный смысл формулы: 2^n двоичных строк длиной n сгруппированы по числу единиц: имеется C_n^k строк с k единицами.

Рассмотрим некоторые свойства сочетаний. Существует ряд тождеств с биномиальными коэффициентами. Некоторыми из них являются следующие:

1. $C_n^m = C_n^{n-m}$ (следует из $C_n^m = \dfrac{n!}{(n-m)!m!}$ и $C_n^{n-m} = \dfrac{n!}{(n-m)!m!}$).

2. $C_n^n = C_n^0 = 1$. Можно составить только одно сочетание из n элементов по n, которое содержит все n элементов. Формула числа сочетаний дает это значение, если учесть, что $0! = 1$ по определению факториала.

3. $C_n^0 + C_n^1 + \ldots + = C_n^n = 2^n$ (следует из бинома Ньютона при $x = y = 1$).

4. $C_n^m = C_{n-1}^{m-1} + C_{n-1}^m$ (следует из определения $C_n^m = \dfrac{n!}{(n-m)!m!}$).

5. Значения C_n^m могут быть найдены с помощью так называемого треугольника Паскаля (французский математик Блез Паскаль (1623–1662)), который имеет вид

$$\begin{array}{c}
1 \\
1\ 2\ 1 \\
1\ 3\ 3\ 1 \\
1\ 4\ 6\ 4\ 1 \\
1\ 5\ 10\ 10\ 5\ 1 \\
1\ 6\ 15\ 20\ 15\ 6\ 1 \\
1\ 7\ 21\ 35\ 35\ 21\ 7\ 1 \\
1\ 8\ 28\ 56\ 70\ 56\ 28\ 8\ 1 \\
.\ \ .\ \ .
\end{array}$$

Первая строчка содержит число сочетаний из $n = 1$ ($C_1^0 = C_1^1 = 1$) элементов, вторая – числа сочетаний из $n = 2$ (C_2^0, C_2^1, C_2^2) элементов, и т. д. Правило построения треугольника: сумма двух рядом стоящих чисел, дает число, стоящее между ними на строке ниже.

3.2 Сложность алгоритмов

Каждый решающий ту или иную практическую задачу сталкивается с проблемой рационального выбора алгоритма для ее решения. Решение проблемы выбора упрощается при наличии системы сравнительных оценок трудоемкости алгоритмов.

3.2.1 Оценка трудоемкости алгоритмов

Алгоритм – это точное предписание, которое определяет процесс, ведущий от исходных данных к требуемому конечному результату. Слово алгоритм происходит от латинской транслитерации *algoritmi* имени арабского математика IX в. аль-Хорезми.

Алгоритм служит, как правило, для решения не одной конкретной задачи, а некоторого класса задач, различающихся исходными данными. Так, алгоритм сложения применим к любой паре натуральных чисел. Следует различать описание алгоритма и процесс его реализации – последовательность шагов, порождаемую при применении к конкретным исходным данным.

Можно выделить ряд свойств, которым должно удовлетворять описание алгоритма:

1) алгоритм применяется к любым данным из заданного набора и выдает результат для каждого набора начальных данных;

2) алгоритм состоит из элементарных шагов и число их конечно;

3) алгоритм является результативным – имеет остановку;

4) последовательность шагов алгоритма является детерминированной: каждый следующий шаг работы однозначно определяется достигнутым состоянием решаемой задачи.

Задача считается *разрешимой*, если существует решающий ее алгоритм. Однако для задачи может существовать не один алгоритм, ее решающий. На практике интерес представляет наиболее «эффективный» с какой-то точки зрения алгоритм решения задачи. Понятие эффективности в широком смысле связано со всеми вычислительными ресурсами (времени и пространства), необходимыми для работы алгоритма. Однако обычно под самым эффективным алгоритмом понимается самый быстрый. Это связано с тем, что ограничение по времени выполнения является доминирующим фактором, определяющим его пригодность для практики. В 50-х гг. XX в. в математике появилось понятие «быстрые алгоритмы» (например, алгоритм Евклида для вычисления наибольшего общего делителя).

Для сравнения по эффективности алгоритмов решения одной и той же задачи вводят такой термин как *трудоемкость* алгоритма, или *временная сложность*, которая отражает затраты времени, требующиеся для работы алгоритма, и оценивается числом некоторых условных элементарных операций, которые необходимо выполнить при решении задачи. Естественно, эта величина зависит от объема исходных данных, который оценивается некоторым параметром. Например, в задачах обработки одномерных массивов это число элементов в массиве; при обработке двумерных массивов это может быть число строк и столбцов; для графа это может быть число вершин или ребер и т. д.

По аналогии с временной сложностью иногда определяют пространственную сложность алгоритма, только здесь говорят не о количестве элементарных операций, а об объеме используемой памяти.

Вычислительная сложность алгоритма является функцией, зависящей от размера исходных данных, и оценивается:

– количеством элементарных операций, выполняемых алгоритмом для решения экземпляра задачи заданного размера;

– некоторой функцией $f(n)$, где n – натуральное число, задающее объем исходных данных.

Следует заметить, что количество элементарных операций, затраченных алгоритмом для решения конкретного экземпляра задачи, зависит не только от размера входных данных, но и от самих данных. Например, иногда количество операций алгоритма сортировки может значительно уменьшиться в том случае, когда входные данные частично отсортированы.

Временная сложность алгоритма определяется временем (затрачиваемым алгоритмом) как функцией размера задачи, и в качестве ее меры принимается сложность:

– в худшем случае (наибольшее значение);

– в наилучшем случае (наименьшее значение);

– в среднем случае (усредненное значение).

Вычисление среднего числа элементарных операций, выполняемых алгоритмом для решения задачи над входными данными заданного размера, является существенно более сложной задачей, чем вычисление такой оценки для наихудшего случая. При анализе сложности алгоритма, как правило, имеется в виду ее оценка в худшем случае, которая определяется как максимальное время, которое требуется для решения задачи для любых исходных данных одного и того же размера n. Эта оценка позволяет также оценить размер задачи, которую можно решить с помощью алгоритма.

Одним из упрощенных видов анализа сложности алгоритмов в худшем случае является асимптотический метод определения порядка ее роста, который связан с поведением алгоритма на входных данных большого размера. Асимптотические оценки позволяют показать скорость роста функции сложности. Для оценивания трудоемкости алгоритмов была введена специальная система обозначений, называемая O-нотацией.

Оценка O представляет собой верхнюю асимптотическую оценку трудоемкости алгоритма. Говорят, что трудоемкость $f(n)$ алгоритма равна $O(g(n))$, если найдется такая константа $c > 0$ и число n_0, что $f(n) \le cg(n)$ для любого $n \ge n_0$. Запись $f(n) = O(g(n))$ означает, что $f(n)$ принадлежит классу функций, которые растут не быстрее, чем функция $g(n)$ с точностью до постоянного множителя. При этом употребляют такие выражения: «трудоемкость алгоритма есть $O(g(n))$» или «алгоритм решает задачу за время $O(g(n))$».

Например, тот факт, что некоторый алгоритм имеет сложность $O(n^3)$, означает, что решение, по крайней мере, одной задачи размера n с помощью этого алгоритма потребует выполнения порядка n^3 операций.

Таким образом, O-нотация позволяет учитывать в функции сложности лишь наиболее значимые элементы, отбрасывая второстепенные, несущественные. $O(g(n))$ описывает характер поведения функции $f(n)$ с ростом n: насколько быстро или медленно растет эта функция. Эта оценка позволяет разбить все основные функции на ряд групп в зависимости от скорости их роста.

Если трудоемкость алгоритма ограничена значением, не зависящим от размера исходных данных, то для ее обозначения используется символ $O(1)$. Алгоритм трудоемкости $O(1)$ называется алгоритмом постоянного времени. Следует заметить, что время работы такого алгоритма не обязательно должно не зависеть от размера исходных данных, тогда как верхняя оценка должна не зависеть.

Алгоритм трудоемкости $O(n)$ называют *линейным*. Алгоритм, для которого трудоемкость $f(n) = O(\log n)$, выполняется за логарифмическое время. В качестве основания степени обычно используется 2 в связи с тем, что при компьютерных вычислениях используется двоичная система счисления. Однако в связи с тем, что логарифмы по разным основаниям отличаются на постоянный множитель (так как пересчет логарифма при изменении основания производится по формуле $\log_a n = \dfrac{\log_b n}{\log_b a}$), которым можно пренебречь при записи верхней оценки трудоемкости алгоритма, то в записи логарифмического роста времени основание логарифма опускается.

Алгоритм трудоемкости $O(n^b)$, где b – некоторая константа (возможно, дробная), называется *полиномиальным*. Время работы (или число выполняемых элементарных операций) такого алгоритма ограничено сверху некоторым полиномом $P(n)$. Если время работы алгоритма растет не быстрее, чем функция $g(n)$, которая является показательной функцией a^n, где a – константа (например, $g(n) = 2^n$), то говорят, что алгоритм имеет *неполиномиальную*, или *экспоненциальную*, сложность.

Задачи, для решения которых известен алгоритм только экспоненциальной сложности $O(a^n)$ (полиномиальный алгоритм не известен), считаются *трудно решаемыми*. Это утверждение исходит из того, что экспоненциальная функция растет быстрее полиномиальной для больших значений n. Однако не все и полиномиальные алгоритмы обеспечивают решение задачи за приемлемое время. Например, если сложность алгоритма равна $O(n^b)$ и значение константы b велико (например, 100 и более), то трудно ожидать, что с его помощью можно будет получить решение задачи за сколько-нибудь разумное время для входных данных большого размера.

3.2.2 Сравнение скорости роста временной сложности

О темпах роста временной сложности выполнения алгоритмов разной трудоемкости можно судить по следующим данным о ее росте при удвоении значения основного параметра n задачи:

– $O(1)$ – время работы не изменяется;

– $O(\log\log n)$ – очень медленный рост времени работы;

– $O(\log n)$ – наблюдается логарифмический рост времени работы, время работы увеличивается на величину, близкую к константе;

– $O(n)$ – наблюдается линейный рост времени работы, время работы удваивается;

– $O(n\log n)$ – наблюдается линеаритмичный рост времени работы, время работы увеличивается чуть более чем вдвое;

– $O(n^2)$ – наблюдается квадратичный рост времени работы, время работы увеличивается в четыре раза;

– $O(n^3)$ – наблюдается кубичный рост времени работы, время работы увеличивается в восемь раз;

– $O(c^n)$ – наблюдается экспоненциальный рост времени работы, время работы увеличивается в квадрат.

Оценка трудоемкости алгоритма позволяет также судить о том, как влияет повышение быстродействия вычислительной машины на время выполнения алгоритма и насколько сложную задачу можно выполнить на ней за фиксированное время. Пусть имеется несколько алгоритмов с разной трудоемкостью и пусть единицей измерения трудоемкости алгоритма является условная элементарная операция. В таблице 3.1 приведены размеры задач, которые могут быть решены каждым из упомянутых в ней алгоритмов за одну секунду, одну минуту и один час. Из этой таблицы видно, например, что за одну минуту алгоритм с трудоемкостью n^2 решает задачу в шесть раз большую, чем алгоритм с трудоемкостью n^3.

В таблице 3.2 приведено время работы алгоритмов разной сложности при увеличении размерности n входных данных. Следует заметить, что $n! \geq 2^n$ (для $n \geq 4$).

Таблица 3.1 – Связь трудоемкости алгоритма с максимальным размером задачи, решаемой за единицу времени

Трудоемкость алгоритма	Максимальный размер задачи		
	1 с	1 мин	1 ч
n	1000	6×10^4	$3{,}6 \times 10^6$
$n \log n$	140	4893	$2{,}0 \times 10^5$
n^2	31	244	1897
n^3	10	39	153
2^n	9	15	21

Таблица 3.2 – Время работы алгоритмов разной сложности на компьютере, выполняющем 1 000 000 оп/с

Сложность алгоритма	n = 10	n = 20	n = 30	n = 40	n = 50	n = 60
n	0,00001 с	0,00002 с	0,00003 с	0,00004 с	0,00005 с	0,00006 с
n^2	0,0001 с	0,0004 с	0,0009 с	0,0016 с	0,0025 с	0,0036 с
n^5	0,1 с	3,2 с	24,3 с	1,7 мин	5,2 мин	13 мин
2^n	0,001 с	1 с	17,9 мин	12,7 дней	35,7 лет	366 ст.
3^n	0,059 с	58 мин	6,5 лет	3855 ст.	$2 \cdot 10^8$ ст.	$1{,}3 \cdot 10^{13}$ ст.
$n!$	3,6 с	771,5 ст.	$8 \cdot 10^{16}$ ст.	…	…	…

3.2.3 Классы сложности алгоритмов

В теории алгоритмов класс сложности образует множество задач, для решения которых известны алгоритмы, схожие по вычислительной сложности. Приведем два основных важных класса задач.

Класс **P** включает все те задачи, для которых известны алгоритмы, решающие их за время, которое ограничено сверху некоторым полиномом от размера входных данных. Задачи класса **P** могут быть решены за полиномиально ограниченное время с помощью детерминированной вычислительной машины.

Например, машины Тьюринга, следующее состояние которой однозначно определяется предыдущим. К этому классу относится, например, сортировка, поиск в множестве, проверка связности графов и многие другие.

Задача, для которой доказано, что время работы каждого из известных решающих ее алгоритмов имеет экспоненциальную сложность, не входит в класс **P**.

Класс **NP** включает задачи, которые могут быть решены за полиномиальное время только с помощью (абстрактной) недетерминированной вычислительной машины. Такая машина для любого своего состояния допускает одновременное выполнение более одного процесса. Детерминированный алгоритм, доходит до состояния, в котором нужно выбрать одну из нескольких альтернатив выполнения. Недетерминированность означает, что алгоритм, начиная с этого состояния, исследует все эти возможности одновременно, как бы копируя самого себя для каждой альтернативы. Все эти копии работают независимо (и могут образовывать новые копии) до тех пор, пока одна из них не найдет решение. Детерминированный алгоритм в отличие от недетерминированного всегда исследует только одну из нескольких альтернатив и, только закончив ее анализ, возвращается для выбора следующей.

Поскольку детерминированная вычислительная машина может рассматриваться как специальный случай недетерминированной, то класс **NP** включает в себя класс **P**, а также некоторые проблемы, для решения которых известны лишь алгоритмы распознавания, трудоемкость которых экспоненциально зависит от размера входа.

В класс **NP** входят многие знаменитые проблемы, такие как выполнимость КНФ (конъюнктивной нормальной формы), задача о ранце (или об укладке рюкзака), задача о коммивояжере, а также многие задачи теории графов (например, раскраски, поиска наименьшего вершинного покрытия, наибольшего независимого множества, клики, гамильтонова цикла) и др. И вообще, для многих практических комбинаторных задач известны алгоритмы только экспоненциальной сложности.

Из определения классов **P** и **NP** сразу вытекает следствие: $\mathbf{P} \subseteq \mathbf{NP}$. Однако до сих пор ничего не известно о строгости этого включения, т. е. существует ли задача, принадлежащая **NP**, но не принадлежащая **P**. Если такой задачи не существует, то все задачи, принадлежащие классу **NP**, можно будет решать за полиномиальное время, что существенно повысит скорость вычислений. Вопрос о равенстве (или неравенстве) этих классов **P** и **NP** считается одной из самых сложных проблем в области теоретической информатики. Математический институт Клэя включил эту проблему в список проблем тысячелетия, предложив награду размером в один миллион долларов США за ее решение.

3.2.4 Вычисление оценок трудоемкости алгоритмов

Рассмотрим два правила оценки временной сложности программы, реализующей несколько алгоритмов, каждый из которых решает свою подзадачу. Два алгоритма, реализующие фрагменты программы, могут выполняться последовательно или один из них может быть вложен в другой. При последовательном выполнении алгоритмов общая трудоемкость определяется трудоемкостью алгоритма с большей оценкой временной сложности. При вложенном выполнении алгоритмов общая трудоемкость определяется произведением их оценок. Таким образом, при вычислении оценок временной сложности алгоритмов следует учитывать следующие правила.

Правило 1 суммы. Пусть $F_1(n)$ и $F_2(n)$ – время выполнения двух последовательных фрагментов программы P_1 и P_2 соответственно, и пусть $F_1(n) = O(g_1(n))$, $F_2(n) = O(g_2(n))$. Тогда $F_1(n) + F_2(n) = O(\max(g_1(n), g_2(n)))$.
Из правила суммы следует, что если $g(n) \le h(n)$ для всех $n \ge n_0$, то $O(g(n)) + O(h(n)) = O(h(n))$. Из этого правила также можно заключить, что можно пренебречь слагаемыми, имеющими меньший порядок роста. Например, $O(n^3+100n) = O(n^3)$.

Правило 2 произведений. Пусть $F_1(n)$ и $F_2(n)$ – время выполнения двух вложенных фрагментов программы P_1 и P_2 соответственно, и пусть $F_1(n) = O(g_1(n))$, $F_2(n) = O(g_2(n))$. Тогда $F_1(n)\, F_2(n) = O(g_1(n))\, O(g_2(n))$.

Из этих правил следует, что при анализе сложности алгоритмов оценки трудоемкости вычисляют с точностью до порядка, пренебрегая слагаемыми меньших порядков и постоянными множителями. Например, $O(100\, g(n)) = O(g(n))$, $O(5n^4 + 4n^2 + 10) = O(n^4)$.

Рассмотрим некоторые примеры вычисления оценок временной сложности алгоритмов.

1. Задача перемножения матриц. Результатом умножения матрицы $\boldsymbol{A}$, имеющей размер $m \times n$ (m, n – числа строк и столбцов), и $\boldsymbol{B}$, имеющей размер $n \times k$, является матрица $\boldsymbol{C}$, имеющая размер $m \times k$, каждый элемент c_{ij} которой равен сумме произведений элементов в i-й строке матрицы $\boldsymbol{A}$ и j-м столбце матрицы $\boldsymbol{B}$: $c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$. Алгоритм умножения матриц, основанный на таком определении произведения матриц, требует $m \times n \times k$ операций умножения. От-

сюда получается кубическая сложность такого алгоритма перемножения матриц $O(n^3)$.

2. Задачи сортировки. Требуется упорядочить по неубыванию (невозрастанию) массив из n элементов, на котором задано отношение частичного порядка «≤». Одним из простейших алгоритмов является сортировка пузырьком, при которой совершается несколько проходов по массиву. При каждом проходе попарно сравниваются соседние элементы и при необходимости (если первый элемент больше второго) меняются местами. В результате первого прохода максимальный элемент окажется в конце массива. Проход по массиву повторяется. Если в результате прохода произошел хотя бы один обмен, проход по массиву повторяется. В худшем случае потребуется до n проходов, в каждом из которых выполнится до n сравнений. По правилу произведений сложность алгоритма сортировки пузырьком равна $O(n^2)$.

3. Задача об укладке рюкзака (задача о ранце). Имеется рюкзак объемом V и n предметов, для каждого из которых известна стоимость c_i и занимаемый объем v_i. Определить, какие предметы необходимо уложить в рюкзак так, чтобы их суммарный объем не превышал допустимый объем V рюкзака: $\sum_i v_i \le V$, а их общая стоимость $C = \sum_i c_i$ была наибольшей.

Для задачи о ранце не известен полиномиальный алгоритм. Простейший алгоритм ее решения сводится к перебору всех вариантов загрузки рюкзака. При условии, что каждый из n предметов может быть упакован или не упакован в рюкзак, всего имеется 2^n вариантов выбора предметов. Соответственно временная сложность такого алгоритма анализа всевозможных вариантов укладки рюкзака равна 2^n.

4. Задача о коммивояжере. Даны n городов и известны расстояния между каждыми двумя городами. Коммивояжер, выходящий из некоторого города, должен посетить $n-1$ других городов (по одному разу каждый) и вернуться в исходный. Определить, в каком порядке ему нужно посещать города, чтобы общее пройденное расстояние было минимальным. Для этой задачи также не известен полиномиальный алгоритм. Простейший переборный алгоритм решения задачи коммивояжера сводится к анализу всех возможных маршрутов, а их всего существует $(n-1)!$.

3.3 Методы комбинаторного поиска

3.3.1 Особенности комбинаторных задач

Для очень многих комбинаторных задач дискретной математики известны алгоритмы только экспоненциальной трудоемкости. Трудоемкость их решения существенно не снижается и с совершенствованием вычислительной техники, сопровождаемой ростом быстродействия вычислительных машин. Косвенным подтверждением этого утверждения являются данные, приведенные в таблице 3.3. В таблице 3.3 показано, как с увеличением быстродействия вычислительных машин в десять раз будут возрастать размеры задач, которые могут быть решены за некоторую фиксированную единицу времени.

Таблица 3.3 – Связь размера задачи, решаемой за заданное время, с быстродействием вычислительной машины

Временна́я сложность	Максимальный размер задачи	
	до ускорения	после ускорения
n	s_1	$10\ s_1$
$n \log n$	s_2	$\approx 10\ s_2$
n^2	s_3	$3{,}16\ s_3$
n^3	s_4	$2{,}15\ s_4$
2^n	s_5	$s_5 + 3{,}3$

Из этой таблицы видно, что задачи достаточно большого размера, решаемые только алгоритмами экспоненциальной трудоемкости, не могут быть решены за практически приемлемое время даже с существенным увеличением быстродействия вычислительных машин.

Комбинаторные задачи характеризуются тем, что поиск их решения связан с перебором элементов некоторого обширного множества. Зачастую по условиям задачи необходимо найти наилучшее в каком-то смысле решение. Ввиду того что множество, среди элементов которого отыскивается решение, всегда конечно, реализация полного перебора вариантов приводит к получению решения, если оно не существует. Таким образом, всякая подобная задача может быть решена за конечное время. Однако это не означает, как уже говорилось выше, что она может быть решена за практически приемлемое время даже с помощью самой быстродействующей вычислительной машины.

Исходя из данных таблицы 3.2 можно проследить, как растут функции сложности алгоритмов экспоненциальной сложности с ростом размера исходных данных и при каких значениях параметров исходных данных еще можно получить решение задачи. Иногда «потолок» решения комбинаторной задачи можно поднять путем сокращения объема перебора на основе учета некоторой дополнительной информации о ней и свойствах ее решений.

Другим выходом из такого положения для комбинаторных задач, связанных с получением лучшего решения, является использование приближенных алгоритмов, не гарантирующих получение оптимального (точного) решения задачи (например, минимум пути обхода коммивояжера), но дающих решение, близкое к оптимальному.

3.3.2 Дерево поиска

Самый общий подход к решению комбинаторных задач основан на обходе дерева поиска, которое строится в процессе их решения. Процесс имеет рекурсивный характер: на каждом шаге текущая задача заменяется несколькими подобными, но меньшего размера. Полученные задачи рассматриваются затем в некотором порядке, по возможности упрощаются, причем менее перспективные могут отбрасываться. При обходе дерева поиска чередуются процессы декомпозиции возникающих ситуаций и их редуцирования, причем естественно стремление к декомпозиции на возможно меньшее число более простых ситуаций и к возможно более глубокому редуцированию. Методы минимизации дерева поиска существенно зависят от специфики конкретных задач.

Корень дерева поиска ставится в соответствие исходной ситуации в процессе решения задачи, остальные вершины сопоставляются с ситуациями, которые можно достичь в данном процессе. Дуги дерева соответствуют некоторым простым операциям, представляющим шаги процесса решения, и связывают вершины, соответствующие ситуациям, одна из которых преобразуется в другую в результате выполнения одного шага. Решения рассматриваемой задачи могут представляться некоторыми вершинами или путями. Иногда требуется найти все решения, иногда – одно из них, любое или оптимальное в некотором смысле. Дерево поиска не задается априори, а строится в процессе поиска: когда возникает некоторая ситуация, тогда и определяются возможные направления процесса, которые представляются дугами, исходящими из вершины, соответствующей данной ситуации. Естественным является стремление сократить число этих дуг, чтобы быстрее найти решение. Способы этого сокращения строятся с учетом особенностей конкретных задач.

Процесс обхода дерева поиска демонстрируется ниже на примере решения задачи о кратчайшем покрытии булевой матрицы, к которой сводятся многие комбинаторные оптимизационные задачи дискретной математики.

3.3.3 Задача о кратчайшем покрытии

Задача о кратчайшем покрытии формулируется следующим образом. Пусть даны некоторое множество $A = \{a_1, a_2, \dots, a_n\}$ и его покрытие $B = \{B_1, B_2, \dots, B_m\}$ – совокупность подмножеств $B_i \subseteq A$, таких, что $B_1 \cup B_2 \cup \dots \cup B_m = A$. Требуется найти минимальное по мощности подмножество $B_{\min} = \{B^k{}_1, B_2{}^k, \dots, B_p{}^k\} \subseteq B$, которое является покрытием множества A, т. е. $B_i^k \subseteq A$ (или $B_i^k \in B$) и $B_1{}^k \cup B_2{}^k \cup \dots \cup B_p{}^k = A$. Другими словами, требуется удалить из множества B максимальное количество подмножеств B_i, сохранив условие покрытия множества A.

Одной из интерпретаций этой задачи является задача о переводчиках. Из m переводчиков, каждый из которых владеет несколькими языками из n заданных, требуется скомплектовать минимальную по числу членов группу, такую, чтобы она смогла обеспечить перевод с любого из заданных n языков. Здесь A – множество языков, перевод с которых требуется обеспечить, а B_i – множество языков, которыми владеет i-й переводчик.

Удобно рассматривать матричную формулировку данной задачи, при которой совокупность $B_1, B_2, \dots, B_m$ задается в виде двоичной или *булевой* матрицы $\boldsymbol{B}$, каждый элемент которой имеет значение 0 или 1. Столбцам этой матрицы поставим в соответствие элементы множества A, а строкам – подмножества $B_i \in B$. Элемент $b_i^j \in \boldsymbol{B}$, на пересечении i-й строки и j-го столбца, имеет значение 1, если $a_j \in B_i$, и 0 – в противном случае. Если $b_i^j = 1$, то говорят, что i-я строка *покрывает* j-й столбец. По условию задачи требуется найти такое множество строк матрицы $\boldsymbol{B}$, чтобы каждый ее столбец имел единицу хотя бы в одной строке из этого множества, и при этом мощность выбранного множества должна быть минимальной.

Можно предложить *приближенные методы* решения задачи покрытия булевой матрицы. Например, ее можно решать с помощью жадного алгоритма, представляющего собой многошаговый процесс, где на каждом шаге выбирается и включается в покрытие та строка заданной матрицы, которая покрывает наибольшее число из еще не покрытых столбцов. Процесс выбора строк заканчивается, когда все столбцы матрицы оказываются покрытыми. Применение жадного алгоритма иногда может дать точное решение, но не гарантирует это.

Например, пусть требуется найти кратчайшее строчное покрытие следующей булевой матрицы:

$$\boldsymbol{B} = \begin{array}{cccccccccccc} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} & \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & B_1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & B_2 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & B_3 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & B_4 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & B_5 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & B_6 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & B_7 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & B_8 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & B_9 \end{array}$$

Первой для включения в формируемое решение жадный алгоритм выберет строку B_9, после чего непокрытыми останутся столбцы a_2, a_4, a_6, a_8 и a_9:

$$\boldsymbol{B'} = \begin{array}{cccccc} a_2 & a_4 & a_6 & a_8 & a_9 & \\ 0 & 1 & 0 & 1 & 0 & B_1 \\ 1 & 0 & 1 & 0 & 0 & B_2 \\ 1 & 0 & 0 & 1 & 0 & B_3 \\ 0 & 1 & 0 & 1 & 0 & B_4 \\ 0 & 1 & 0 & 0 & 0 & B_5 \\ 0 & 0 & 1 & 0 & 1 & B_6 \\ 1 & 0 & 0 & 0 & 0 & B_7 \\ 0 & 0 & 0 & 1 & 1 & B_8 \end{array}$$

Для покрытия оставшихся столбцов в решение будут последовательно включены строки B_1, B_2 и B_6. Соответственно найденное покрытие включает четыре строки: B_9, B_1, B_2 и B_6.

Более близкое к кратчайшему покрытию позволяет найти в большинстве случаев «*минимаксный алгоритм*».

Он исходит из очевидных требований к искомому решению: каждый элемент $a_i \in A_i$ должен войти хотя бы в одно из подмножеств B_j, составляющих решение задачи покрытия. Отсюда, если некоторый столбец матрицы имеет единственную единицу, например в i-й строке, то соответствующее множество B_i должно войти в решение, и столбец с меньшим числом единиц имеет меньше возможностей быть покрытым. Минимаксный алгоритм представляет собой многошаговый процесс, на каждом шаге которого сначала выбирается столбец с минимальным числом единиц и из покрывающих его строк для включения в решение выбирается та, которая покрывает максимальное число не покрытых до текущего шага столбцов.

Одним из столбцов приведенной выше матрицы $\boldsymbol{B}$, имеющих минимальное число единиц (две единицы) является столбец a_6. Из покрывающих его строк максимальное число столбцов (четыре) покрывает строка B_6. Включив эту строку в решение и удалив ее и столбцы (a_1, a_3, a_6 и a_9), которые она покрывает, получим текущий непокрытый остаток матрицы $\boldsymbol{B}$:

a_2	a_4	a_5	a_7	a_8	a_{10}	
0	1	0	0	1	0	B_1
1	0	0	1	0	0	B_2
1	0	1	0	1	0	B_3
0	1	0	0	1	1	B_4
0	1	0	1	0	0	B_5
1	0	1	1	0	0	B_7
0	0	1	0	1	0	B_8
0	0	1	1	0	1	B_9

Из оставшихся столбцов минимальное число единиц (два) имеет столбец a_{10}. Покрывающие его строки B_4 и B_9 имеют одинаковое число единиц (три), поэтому включаем в решение первую по порядку строку B_4 и получаем матрицу

a_2	a_5	a_7	
0	0	0	B_1
1	0	1	B_2
1	1	0	B_3
0	0	1	B_5
1	1	1	B_7
0	1	0	B_8
0	1	1	B_9

В полученной матрице столбцом с минимальным числом единиц (три) является столбец a_2, а из покрывающих его строк строка B_7 имеет максимальное число (три) единиц. Включение этой строки в решение завершает процесс, в результате которого получается покрытие $\{B_4, B_6, B_7\}$. Ниже будет показано, что это решение является точным.

Точный метод. Гарантию того, что полученное покрытие действительно будет кратчайшим, может дать лишь алгоритм комбинаторного поиска, в основе которого лежит перебор вариантов, упорядочиваемый обходом дерева поиска.

Текущая ситуация, соответствующая некоторой вершине дерева поиска, представляется переменной матрицей $\boldsymbol{X}$, которая показывает, какие столбцы еще не покрыты и какие строки можно использовать для их покрытия. Начальное значение матрицы $\boldsymbol{X}$ совпадает с исходной матрицей $\boldsymbol{B}$.

В каждой текущей ситуации выбирается первый из столбцов с минимальным числом единиц с тем, чтобы минимизировать число вариантов продолжения поиска. Очередной шаг обхода дерева поиска состоит в выборе покрывающей строки для этого столбца и пробном включении ее в получаемое решение. Таким образом, вершины дерева поиска соответствуют некоторым столбцам исходной матрицы, а дуги – выбираемым для их покрытия строкам. Последующие действия состоят в удалении строк, включаемых в решение, и столбцов, покрытых этими строками.

Существенное сокращение перебора обеспечивается на основе следующих *правил редукции* переменной матрицей, выполняемых в каждой текущей ситуации.

1. *Правило удаления столбца*. Из матрицы $\boldsymbol{X}$ удаляется столбец a_i, если он поглощает некоторый другой столбец a_j, т. е. a_i имеет единицы везде, где имеет единицы столбец a_j. Любая строка, покрывающая столбец a_j, покрывает также столбец a_i, поэтому условие покрытия столбца a_i удовлетворяется при выполнении условия покрытия столбца a_j.

2. *Правило удаления строки*. Из матрицы $\boldsymbol{X}$ удаляется строка B_i, если она поглощается некоторой другой строкой B_j, т. е. B_j имеет единицы везде, где имеет единицы строка B_i. Правило обосновывается тем, что строка B_j обеспечивает покрытие всех столбцов, покрываемых строкой B_i, и еще некоторых, не входящих в это число. Отказ от рассмотрения строки B_i оправдан в тех случаях, когда не ставится задача найти все кратчайшие покрытия.

Продемонстрируем описанный процесс на вышеприведенной матрице:

$$\boldsymbol{B} = \begin{array}{cccccccccc|c}
a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} & \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & B_1 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & B_2 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & B_3 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & B_4 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & B_5 \\
1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & B_6 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & B_7 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & B_8 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & B_9
\end{array}$$

На первом шаге выбирается столбец a_6, содержащий две единицы. Среди покрывающих его строк выбирается строка B_6, которая покрывает наибольшее число столбцов. Удалив эту строку и покрываемые ею столбцы, получим следующее значение матрицы $\boldsymbol{X}$:

$$
\boldsymbol{X} = \begin{array}{cccccc|c}
a_2 & a_4 & a_5 & a_7 & a_8 & a_{10} & \\
0 & 1 & 0 & 0 & 1 & 0 & B_1 \\
1 & 0 & 0 & 1 & 0 & 0 & B_2 \\
1 & 0 & 1 & 0 & 1 & 0 & B_3 \\
0 & 1 & 0 & 0 & 1 & 1 & B_4 \\
0 & 1 & 0 & 1 & 0 & 0 & B_5 \\
1 & 0 & 1 & 1 & 0 & 0 & B_7 \\
0 & 0 & 1 & 0 & 1 & 0 & B_8 \\
0 & 0 & 1 & 1 & 0 & 1 & B_9
\end{array}
$$

В матрице $\boldsymbol{X}$ отсутствуют столбцы, удовлетворяющие правилу удаления столбца, но есть строки B_1, B_2 и B_8, удовлетворяющие правилу удаления строки. После удаления этих строк матрица $\boldsymbol{X}$ будет иметь следующий вид:

$$
\boldsymbol{X} = \begin{array}{cccccc|c}
a_2 & a_4 & a_5 & a_7 & a_8 & a_{10} & \\
1 & 0 & 1 & 0 & 1 & 0 & B_3 \\
0 & 1 & 0 & 0 & 1 & 1 & B_4 \\
0 & 1 & 0 & 1 & 0 & 0 & B_5 \\
1 & 0 & 1 & 1 & 0 & 0 & B_7 \\
0 & 0 & 1 & 1 & 0 & 1 & B_9
\end{array}
$$

Одним из столбцов, обладающих минимальным числом единиц, является столбец a_2. Обе покрывающие его строки B_3 и B_7 содержат по три единицы. Выбирая первую по порядку строку B_3 и включая ее в формируемое покрытие, в качестве текущего решения имеем множество $\{B_3, B_6\}$. Этот шаг приводит к матрице

$$
\boldsymbol{X} = \begin{array}{ccc|c}
a_4 & a_7 & a_{10} & \\
1 & 0 & 1 & B_4 \\
1 & 1 & 0 & B_5 \\
0 & 1 & 0 & B_7 \\
0 & 1 & 1 & B_9
\end{array}
$$

После удаления строки B_7 по правилу удаления строки получается матрица, каждая строка и каждый столбец которой содержат ровно по две единицы. Выбрав строку B_4, покрывающую столбец a_4, и проведя аналогичные преобразования, получим матрицу с одним столбцом a_7 и двумя строками B_5 и B_9, любая из которых покрывает оставшийся столбец. Таким образом, получено текущее покрытие $\{B_3, B_4, B_5, B_6\}$ матрицы $\boldsymbol{B}$, но пройдена пока только одна ветвь дерева поиска, и до совершения полного обхода дерева не известно, является ли это покрытие кратчайшим.

Возвращаемся к ситуации, когда очередным столбцом для покрытия взят a_2. Теперь вместо строки B_3 возьмем для покрытия столбца a_2 строку B_7. Действуя дальше аналогичным образом, получаем очередное покрытие $\{B_4, B_6, B_7\}$, которое вытесняет предыдущее, так как оно оказалось лучше, однако и его пока нельзя назвать кратчайшим.

Возвратившись к начальной вершине дерева поиска и следуя по дуге, соответствующей строке B_2, убеждаемся, что длина покрытия не может быть меньше трех. На этом поиск можно закончить и выдать в качестве решения множество $\{B_4, B_6, B_7\}$. Дерево поиска, обход которого совершался в процессе решения данного примера, приведено на рисунке 3.1.

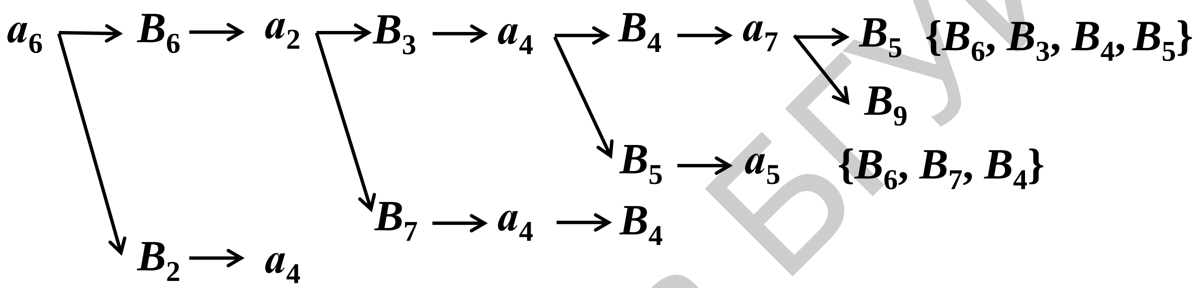


Рисунок 3.1 – Дерево поиска кратчайшего покрытия

Задания

1. Сколько существует наборов длиной n из нулей и единиц?

2. Сколькими способами можно составить набор из карандаша, тетради и резинки, если имеется: 12 карандашей, 5 тетрадей и 3 резинки?

3. Сколькими способами можно выбрать шестерых человек для дежурства из группы 25 студентов?

4. Сколько различных костей имеет домино?

5. Сколько различных 5-буквенных слов можно составить из символов a, a, b, c, d?

6. Сколько перестановок можно получить из букв слова КРОКОДИЛ?

7. Сколькими способами можно выбрать 5 конфет из имеющихся конфет трех сортов, если имеется 6 конфет первого сорта, 7 – второго и 8 – третьего?

8. Сколькими способами можно разложить 10 палочек по 12 коробкам?

9. Сколькими способами можно разложить 10 палочек по 12 коробкам, но так, чтобы в коробке было не более одной?

10. Номер автомобиля состоит из двух букв и четырех цифр. Сколько различных номеров можно составить, используя 10 цифр и 26 букв?

11. Сколькими способами можно составить четырехзначное число, все цифры которого различны?

12. Сколькими способами можно разместить 20 пассажиров в купе по 4 человека?

13. На рояле 88 клавиш. Сколькими способами можно извлечь последовательно 8 разных звуков?

14. Сколько можно составить пятизначных чисел, в десятичной записи которых хотя бы один раз встречается цифра 3?

15. Сколько можно составить пятизначных чисел, делящихся на 5, в записи которых нет одинаковых цифр?

16. Сколькими способами можно поставить в ряд 10 девочек и 10 мальчиков так, чтобы: а) никакие две девочки (и никаких два мальчика) не сидели рядом; б) все девочки сидели рядом?

17. Полотно состоит из 12 полос белого, зеленого и красного цвета. Сколькими способами можно чередовать цвета так, чтобы любые соседние полосы имели разный цвет?

18. В магазине имеется 6 белых и 9 красных роз. Сколькими способами можно выбрать две белых и 1 красную розы?

19. Сколькими способами можно разложить 13 эклеров и 7 безе в 2 пакета так, чтобы в каждом пакете было одно и то же число пирожных и хотя бы по одному каждого сорта?

20. Сколькими способами можно расставить в ряд 10 столов так, чтобы A и B не стояли рядом (и стояли рядом)?

21. Сколькими способами можно расставить в ряд числа 1, 2, 3, … n так, чтобы числа 1, 2, 3 стояли друг за другом в порядке возрастания?

22. Сколько пятизначных чисел можно составить из цифр 1, 2, 3, 5, 7, 9, если каждую цифру можно использовать только один раз?

23. Сколькими способами n различных книг, каждая из которых имеется в m экземплярах можно разместить на одной полке?

24. Сколько существует различных булевых функций от n аргументов?

25. Сколькими способами можно n одинаковых подарков раздать r детям ($n \geq r$): а) без ограничений; б) каждый ребенок должен получить хотя бы один подарок?

26. Найти кратчайшее строчное покрытие булевой матрицы:

$$
\boldsymbol{B} = \begin{array}{ccccccccccc|l}
a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} & a_{11} & \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & B_1 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & B_2 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & B_3 \\
1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & B_4 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & B_5 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & B_6 \\
0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & B_7 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & B_8 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & B_9 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & B_{10}
\end{array}
$$

4 ГРАФЫ

Начало теории графов принято относить к 1736 г., когда Леонард Эйлер не только решил популярную тогда задачу о кенигсбергских мостах, но и нашел критерий существования в графе специального маршрута (его называют теперь эйлеровым циклом). Долгое время теория графов имела дело в основном с математическими развлечениями и головоломками. Следующий импульс теория графов получила почти через 100 лет в связи с началом исследований в области электрических сетей, кристаллографии, органической химии и других наук. В середине XIX в. инженер-электрик Г. Кирхгоф разработал теорию деревьев для исследования электрических цепей. Он установил законы, связывающие значения напряжения и тока в цепи. Эта связь для электрической цепи зависит от характера соединений элементов, т. е. от графа, представляющего цепь.

Хотя первая работа по теории графов и принадлежит Л. Эйлеру, сам термин «граф» впервые ввел венгерский математик Денеш Кениг в 1936 г. Графом были названы схемы, состоящие из точек и соединяющих эти точки отрезков прямых или кривых.

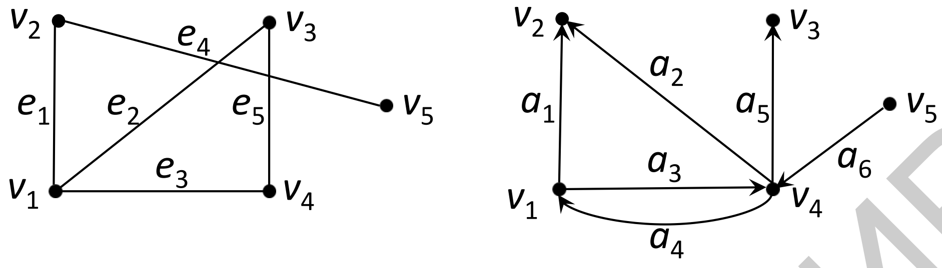
В наше время теория графов находит применение в самых разнообразных областях науки и техники. Многие структуры, представляющие практический интерес в логике, информатике, физике, химии, лингвистике и других областях, могут быть представлены графами, и с их помощью часто упрощается решение различных задач. Теория графов широко используется для решения оптимизационных задач конструирования компиляторов, автоматизации проектирования дискретных устройств, интегральных схем, теории автоматов, проектировании электросетей, планировании транспортных перевозок. Кроме того, в настоящее время теория графов широко используется в экономике и статистике, биологии и социологии.

4.1 Графы: виды и задание

4.1.1 Неориентированный граф

Графом G называется пара множеств (V, E), где V – непустое множество элементов, называемых *вершинами*, и E – некоторое множество пар (v_i, v_j) элементов из множества V, называемых *ребрами*, где $v_i, v_j \in V$ и $E \subseteq V^2$.

Вершины v_i и v_j графа $G = (V, E)$ являются *концами* (концевыми вершинами) ребра $(v_i, v_j) \in E$. Например, концами ребра e_1 графа (рисунок 4.1, *а*) являются вершины v_1 и v_2. Принято обозначать ребра парами их концов, например $e_1 = (v_1, v_2)$ или $e_1 = v_1 v_2$.



а *б*

а – неориентированный граф; *б* – ориентированный граф

Рисунок 4.1 – Примеры графов

Само понятие графа подразумевает его графическое представление. Граф удобно изображать в виде геометрической фигуры, в которой каждая вершина изображается точкой, а каждое ребро представляется отрезком линии, соединяющей его концевые вершины. Для многих задач теории графов не важен вид и длина ребер, существенно лишь то, какие вершины соединяются ребрами.

В практических приложениях рассматриваются *конечные графы*, имеющие конечное число вершин (и соответственно ребер). Множество вершин графа всегда считается непустым, в то время как множество ребер может быть и пустым. Граф $G = (V, E)$ с числом $n = |V|$ вершин и $m = |E|$ ребер называют также (n, m)-графом. Число вершин называется *порядком*, а число ребер – *размером* графа.

Существуют два основных вида графов (и множество их подвидов): ориентированные и неориентированные (рисунок 4.1). Если ребра графа представляют упорядоченные пары вершин (ребрам приданы направления), то граф называется *ориентированным* графом (*орграфом*, направленным графом), а ориентированные ребра – *дугами*. Если порядок вершин в парах (v_i, v_j) не задан (ребрам не приданы направления), то граф называется *неориентированным*.

Вершины v_i и v_j графа $G = (V, E)$ являются *концами* (концевыми вершинами) ребра $(v_i, v_j) \in E$. Например, концами ребра e_1 графа (см. рисунок 4.1, *а*) являются вершины v_1 и v_2. Принято обозначать ребра парами их концов, например $e_1 = (v_1, v_2)$ или $e_1 = v_1 v_2$.

Граф называется *конечным*, если множество V его вершин (а значит, и E ребер) конечно, иначе граф является *бесконечным*. Конечный неориентированный граф в смысле приведенного выше определения называется *простым* (или *обыкновенным*). В таком графе все ребра (или пары (v_i, v_j) вершин) и их концы различны (нет ребер (v_i, v_i). Далее без специальной оговорки будут иметься в виду именно простые графы.

Между вершинами и ребрами графа вводится отношение инцидентности. Говорят, что вершина $v \in V$ и ребро $e \in E$ графа $G = (V, E)$ *инцидентны*, если v является одним из концов ребра e. Две вершины графа называются *смежными* (соседними), если они инцидентны одному и тому же ребру. Аналогично два ребра смежны, если они имеют общую вершину.

Например, вершины v_1 и v_3 графа на рисунке 4.1, *а* являются смежными, каждая из них инцидентна ребру e_2. Аналогично ребра e_2 и e_3 смежны друг другу и инцидентны вершине v_1.

Заметим, что отношение смежности определено между однородными элементами графа (или вершинами, или ребрами), а отношение инцидентности – между разнородными (между вершинами и ребрами).

Множество всех вершин графа G, смежных с вершиной v, называется ее *окрестностью* и обозначается символом Γv. Мощность множества Γv называется *степенью* (или валентностью) вершины v и обозначается через $d(v)$. Другими словами, степенью вершины графа называется число ребер, инцидентных этой вершине. Вершина v, степень которой $d(v) = 0$, называется *изолированной*, а вершина v со степенью $d(v) = 1$ – *висячей*.

Например, $\Gamma v_1 = \{v_2, v_3, v_4\}$ и $d(v_1) = 3$. Граф (см. рисунок 4.1, *а*) имеет одну висячую вершину v_5 и не имеет изолированных вершин.

Для неориентированного графа с множеством ребер E очевидно утверждение (лемма о рукопожатиях), представляемое следующим соотношением:

$$\sum_{v \in V} d(v) = 2|E|,$$

из которой следует, что в любом неориентированном графе число вершин с нечетной степенью всегда четно. В рассматриваемом графе таких вершин две: v_1 с $d(v_1) = 3$ и v_5 с $d(v_5) = 1$.

Поскольку граф можно рассматривать как графическое представление некоторого бинарного отношения на множестве его вершин, то его, так же как и бинарное отношение, можно задать в матричном виде. Этот способ представления графа часто используется при решении задач над графами. Приведем основные способы представления графа $G = (V, E)$.

1. *Графическое представление* графа.

2. *Матрица смежности* графа задает отношение смежности на множестве вершин графа и представляет собой булеву (или бинарную) матрицу, эле-

менты которой равны 0 или 1. Строки и столбцы матрицы смежности соответствуют вершинам графа, а элемент на пересечении строки v_i и столбца v_j имеет значение 1 тогда и только тогда, когда вершины v_i и v_j смежны. Нетрудно видеть, что матрица смежности неориентированного графа симметрична относительно главной диагонали, которая состоит из нулевых элементов.

Заметим, что любая строка матрицы смежности является векторным представлением окрестности соответствующей вершины.

3. *Матрица инцидентности* задает отношение *инцидентн*ости между вершинами и ребрами графа и представляет собой булеву матрицу. Ее строки соответствуют вершинам графа, а столбцы – ребрам, а элемент на пересечении строки v_i и столбца e_j имеет значение 1 тогда и только тогда, когда вершина v_i и ребро e_j инцидентны. Нетрудно заметить, что любой столбец матрицы инцидентности неориентированного графа содержит ровно две единицы.

К примеру, граф, приведенный на рисунке 4.1, *а*, представляется следующими матрицами смежности и инцидентности:

$$\begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & \\ 0 & 1 & 1 & 1 & 0 & v_1 \\ 1 & 0 & 0 & 0 & 1 & v_2 \\ 1 & 0 & 0 & 1 & 0 & v_3 \\ 1 & 0 & 1 & 0 & 0 & v_4 \\ 0 & 1 & 0 & 0 & 0 & v_5 \end{array} \qquad \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & \\ 1 & 1 & 1 & 0 & 0 & v_1 \\ 1 & 0 & 0 & 1 & 0 & v_2 \\ 0 & 1 & 0 & 0 & 1 & v_3 \\ 0 & 0 & 1 & 0 & 1 & v_4 \\ 0 & 0 & 0 & 1 & 0 & v_5 \end{array}$$

4. *Списки смежности* вершин графа. Этот способ задания состоит в указании для каждой вершины v графа ее окрестности Γv – списка вершин, смежных с v.

5. *Список ребер* графа задает пары вершин, связанных ребром.

Например, граф на рисунке 4.1, *а* задается следующим списком смежности вершин и списком ребер:

$$\Gamma v_1 = \{v_2,v_3,v_4\},\ \Gamma v_2 = \{v_1,v_5\},\ \Gamma v_3 = \{v_1,v_4\},\ \Gamma v_4 = \{v_1,v_3\},\ \Gamma v_5 = \{v_2\};$$

$$\{(v_1,v_2),\ (v_1,v_3),\ (v_1,v_4),\ (v_2,v_5),\ (v_3,v_4)\}.$$

Если граф $G = (V, E)$ является *разреженным*, т. е. число его ребер достаточно мало по сравнению с числом вершин, то более эффективными (для решения некоторых задач теории графов), по сравнению с матричным заданием, являются два последних представления.

Однако, хотя представление графа списком ребер является наиболее экономным (по расходам памяти компьютера), оно имеет и существенный недостаток. Если потребуется проверить, содержит ли граф некоторое ребро, то в худшем случае потребуется просмотреть все элементы списка. В этом смысле более быстрый доступ к ребрам графа обеспечивает представление графа матрицей смежности, но оно требует больших затрат памяти компьютера. Выбор того или иного способа представления графа существенно определяется задачей и алгоритмом ее решения. Для повышения быстродействия алгоритма иногда одновременно используется не одно представление графа.

4.1.2 Операции над графами

Поскольку граф можно рассматривать как графическое представление некоторого бинарного отношения, определяемого множеством пар вершин, то к графам, так же как и к бинарным отношениям, применимы теоретико-множественные операции (см. подраздел 2.2.2), определенные для множеств.

Рассмотрим некоторые операции над простыми графами, необходимые для дальнейшего изложения понятий теории графов.

1. *Объединением* графов $G_1 = (V_1, E_1)$ и $G_2 = (V_2, E_2)$ называется граф $H = (V, E)$ (обозначается как $H = G_1 \cup G_2$), для которого $V = V_1 \cup V_2$ и $E = E_1 \cup V_2$ (рисунок 4.2, *в*). Для графов общего вида обычно рассматривается только дизъюнктивное объединение при условии отсутствия в множествах V_1 и V_2 общих вершин: $V_1 \cap V_2 = \varnothing$.

2. *Пересечением* графов $G_1 = (V, E_1)$ и $G_2 = (V, E_2)$, определенных на одном и том же множестве вершин, называется граф $H = (V, E)$ (обозначается как $H = G_1 \cap G_2$), для которого $E = E_1 \cap E_2$ (рисунок 4.2, *г*).

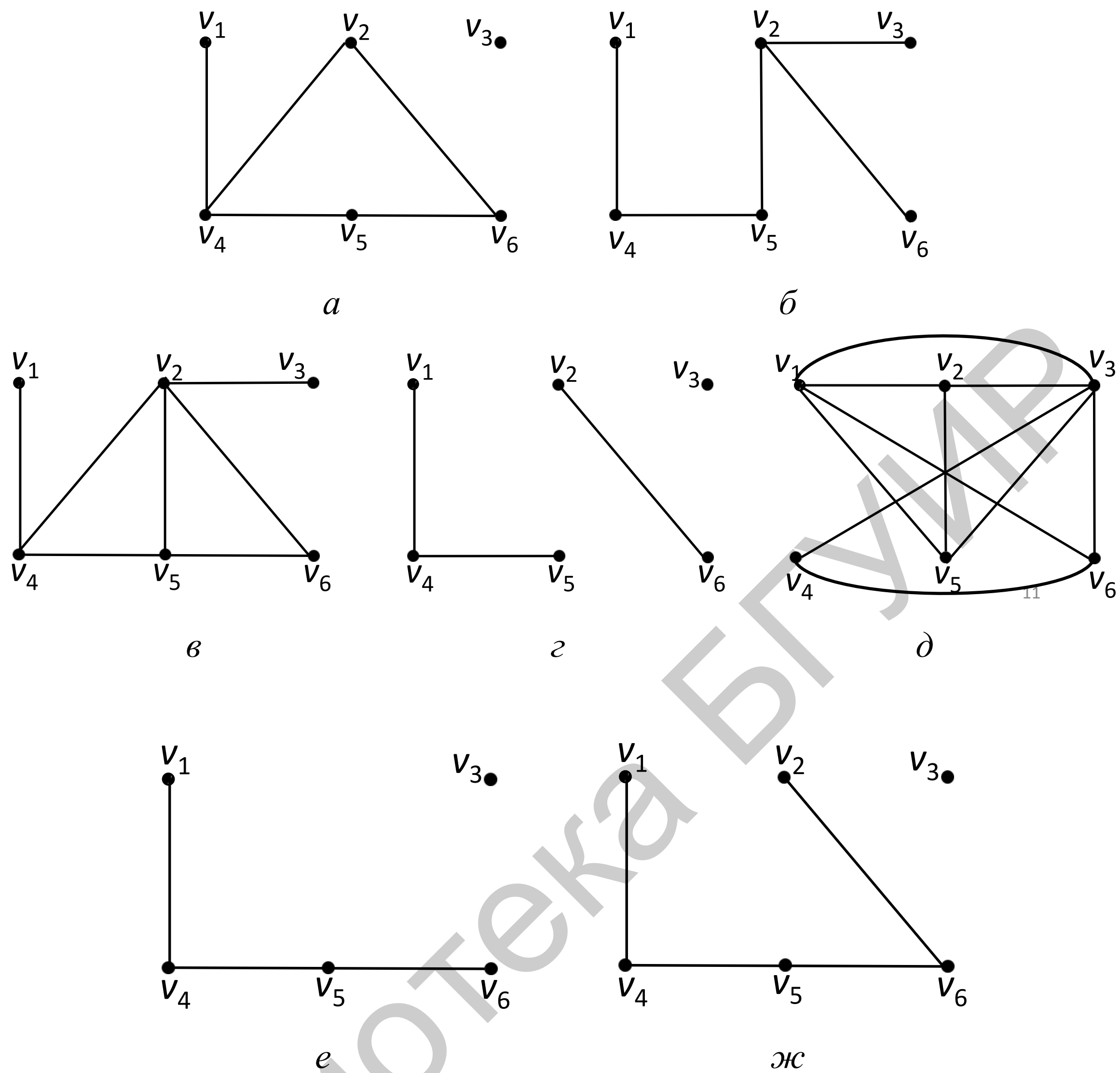
3. *Дополнением* графа $G = (V_1, E_1)$ (обозначаемым как $\overline{G} = (V_1, E_1)$ называется граф $H = (V_2, E_2)$, в котором $V_2 = V_1$ и любые две вершины смежны только тогда, когда они не смежны в графе G. Последнее означает, что E_2 состоит из таких $e \in V_1^2$, что $e \notin E_1$ (рисунок 4.2, *д*).

4. Удаление из графа $G = (V, E)$ вершины $v \in V$ приводит к графу $H = (V', E')$ (обозначается как $H = G - v$), для которого (рисунок 4.2, *е*)

$$V' = V \setminus \{v\},\ E' = E \setminus \{e = (v, v_j)\}.$$

5. Удаление из графа $G = (V, E)$ ребра $e \in E$ дает граф $H = (V', E')$ (обозначается как $H = G - e$), для которого (рисунок 4.2, *ж*)

$$V' = V,\ E' = E \setminus \{e\}.$$



а, *б* – исходные графы G_1 и G_2; *в* – граф $G_1 \cup G_2$; *г* – граф $G_1 \cap G_2$;
д – граф $\overline{G}_1$; *е* – граф $G_1 - v_2$; *ж* – граф $G_1 - (v_2, v_4)$

Рисунок 4.2 – Операции над графами

4.1.3 Специальные типы графов

Граф $G = (V, E)$, у которого множество ребер пусто, т. е. $E = \varnothing$, называется *пустым* графом. Пустой граф является $(n, 0)$-графом (или 0-графом).

Неориентированный граф называется *полным*, если любые две его вершины смежны. Для полного графа, число вершин которого равно n, введено специальное обозначение K_n. Примеры полных графов показаны на рисунке 4.3.

Принимая во внимание, что степени всех вершин полного графа равны $n-1$, согласно лемме о рукопожатиях имеем $n(n-1) = 2m$, т. е. число ребер полного графа равно $m = n(n–1)/2$ (числу сочетаний $C(n, 2)$). Полный граф K_n является $(n, n(n–1)/2)$-графом.

Все элементы матрицы смежности пустого графа имеют значение 0. Матрица смежности полного графа, напротив, содержит нули только на главной диагонали, остальные элементы равны 1.

Очевидно, что полный граф K_n является дополнением пустого n-вершинного графа и, наоборот, $(n, 0)$-граф является дополнением графа K_n.

Граф называется *двудольным*, если множество его вершин V можно разбить на два непересекающихся подмножества V' и V'' так, что концы любого его ребра находятся в различных подмножествах V' и V''. Такой граф задается тройкой $G = (V',V'',E)$.

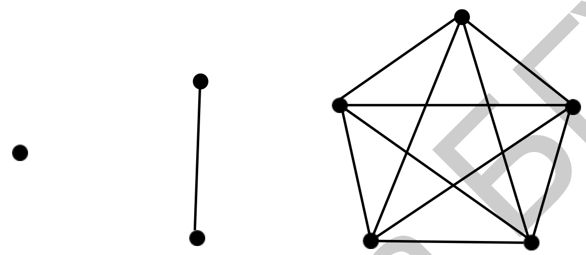
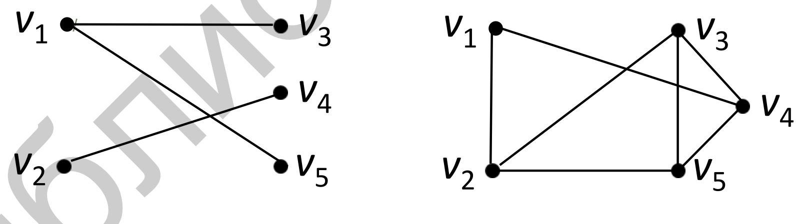


Рисунок 4.3 – Полные графы K_1; K_2 и K_5

Пример двудольного графа $G = (\{v_1,v_2\},\{v_3,v_4,v_5\}, E)$ и его дополнения приведены на рисунке 4.4.



а *б*

а – двудольный граф $G = (\{v_1,v_2\},\{v_3,v_4,v_5\}, E)$; *б* – дополнение графа G

Рисунок 4.4 – Примеры графов

Матрица смежности двудольного графа $G = (V',V'',E)$ может быть представлена в более компактном виде: ее строкам можно поставить в соответствие вершины из V', а столбцам – из V''. Например, матрица смежности двудольного графа $G = (\{v_1,v_2\},\{v_3,v_4,v_5\}, E)$ (рисунок 4.4, а) имеет следующий вид:

$$\begin{array}{ccc|c} v_3 & v_4 & v_5 & \\ 1 & 0 & 1 & v_1 \\ 0 & 1 & 0 & v_2 \end{array}$$

В *полном двудольном* графе (V',V'',E) каждая вершина из V' связана ребром с каждой вершиной из V''. Полный двудольный граф, доли которого состоят из p и q вершин ($|V'| = p$ и $|V''| = q$), обозначается как $K_{p,q}$. Примеры полных двудольных графов показаны на рисунке 4.5.

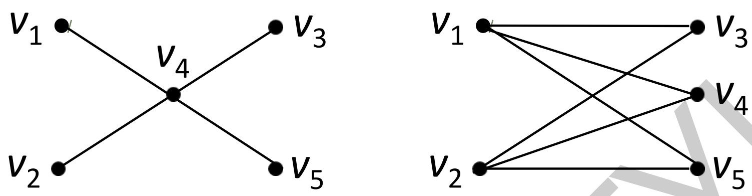


Рисунок 4.5 – Полные двудольные графы $K_{1,4}$ и $K_{2,3}$

По аналогии с двудольным графом определяется *k-дольный* (и полный k-дольный) граф $G = (V_1,V_2,\dots, V_k, E)$, в котором множество вершин V разбито на k непересекающихся подмножеств: $V = V_1 \cup V_2 \cup \dots \cup V_k$, $V_i \cap V_j = \varnothing$ для всех $i \neq j$, и концы любого ребра находятся в различных подмножествах V_1, V_2, …, V_k.

Граф называется *однородным* (или регулярным), если степени всех его вершин равны. В частности, если $d(v_1) = d(v_2) = d(v_3) =\dots = k$, то граф является k-однородным. Степень k вершин называется степенью однородного графа. Например, все полные графы являются однородными.

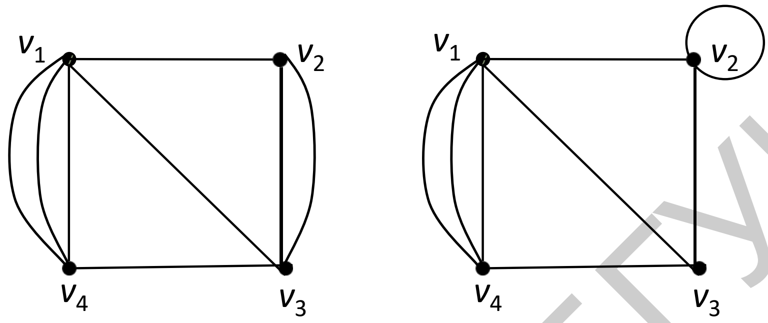
4.1.4 Обобщения графов

Существуют различные обобщения понятия графа. Одним из таких обобщений является *мультиграф*, в нем две вершины могут быть связаны более чем одним ребром. Ребра (v_i, v_j) и (v_k, v_l), у которых концевые вершины совпадают, т. е. $v_i = v_k$, $v_j = v_l$, называются *кратными*. Мультиграфы используются во многих задачах, например: для задания сети дорог между населенными пунктами, для представления молекул в органической химии.

Если в графе допускаются не только кратные ребра, но и *петли*, т. е. ребра (v_i, v_i), соединяющие вершину саму с собой, то такой граф называется *псевдографом*. На рисунке 4.6 приведены мультиграф и псевдограф.

Мультиграфы и псевдографы также имеют матричное представление, однако их матрица смежности не является булевой: для смежных вершин задается число ребер, которыми они связаны. Например, матрица смежности графа на рисунке 4.6, *б* имеет следующий вид:

$$\begin{array}{cccc|c} v_1 & v_2 & v_3 & v_4 & \\ 0 & 1 & 1 & 3 & v_1 \\ 1 & 1 & 1 & 0 & v_2 \\ 1 & 1 & 0 & 1 & v_3 \\ 3 & 0 & 1 & 0 & v_4 \end{array}$$



а *б*

а – мультиграф; *б* – псевдограф

Рисунок 4.6 – Примеры графов

В некоторых задачах используются помеченные и взвешенные графы. Граф называется *помеченным*, если его вершинам назначены некоторые метки, например номера. Метки могут присваиваться также и ребрам (дугам). Пометка отражает семантику области использования графа и служит для идентификации вершин и ребер (дуг). Например, орграф переходов и выходов конечного автомата является помеченным: вершины помечаются символами внутренних состояний, а дуги – входными или выходными состояниями автомата.

В некоторых задачах используется граф, на множестве ребер которого задана функция φ, ставящая в соответствие каждому ребру e положительное (или отрицательное) число φ(e), называемое *весом* ребра. Граф G с определенной на его ребрах функцией φ называется *взвешенным* графом, а точнее – графом с *взвешенными ребрами.* Аналогично определяется граф с *взвешенными вершинами*. Графы с взвешенными ребрами используются в транспортных задачах и в задачах о потоках в сетях. Мультиграф можно рассматривать как граф, ребра которого взвешены натуральными числами, представляющими кратности ребер.

Иногда рассматриваются *смешанные графы*, в которых наряду с дугами (элементами ориентированного графа) имеются ребра (элементы неориентированного графа). В этом случае можно считать, что ребром заменена пара противоположно направленных дуг в ориентированном графе, соединяющих одни и те же вершины. Смешанные графы используются, например, при решении задач, связанных с установлением схемы выполнения операций в технологическом процессе.

Еще одним обобщением понятия графа является *гиперграф*, который также задается парой множеств – множеством вершин и множеством ребер. Однако если ребром графа является пара вершин, то ребром гиперграфа может быть любое непустое подмножество множества вершин. На рисунке 4.7 приведен гиперграф $G = (V, B)$, где $V=\{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$, $B = \{b_1=\{v_1,v_2,v_3\}, b_2=\{v_2,v_4,v_5\}, b_3=\{v_5,v_7\}, b_4=\{v_3,v_7\}, b_5=\{v_6\}\}$.

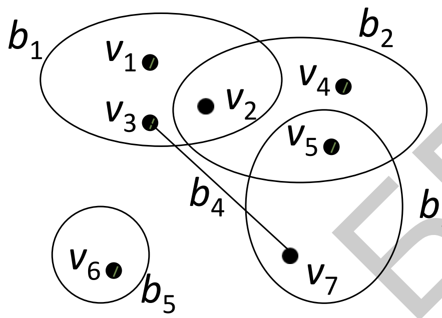


Рисунок 4.7 – Пример гиперграфа

Гиперграф может служить моделью принципиальной электрической схемы. При этом полюса элементов данной схемы соответствуют вершинам гиперграфа, а электрические цепи – ребрам. Электрическая цепь здесь рассматривается как множество выводов, соединенных между собой проводниками. Многие понятия, связанные с графами, распространяются на случай гиперграфа, однако графически изобразить гиперграф гораздо труднее, чем граф. Вместе с тем от гиперграфа можно перейти к двудольному графу, долями которого являются множество вершин и множество ребер гиперграфа, а ребра показывают принадлежность вершин гиперграфа его ребрам.

4.1.5 Части графов

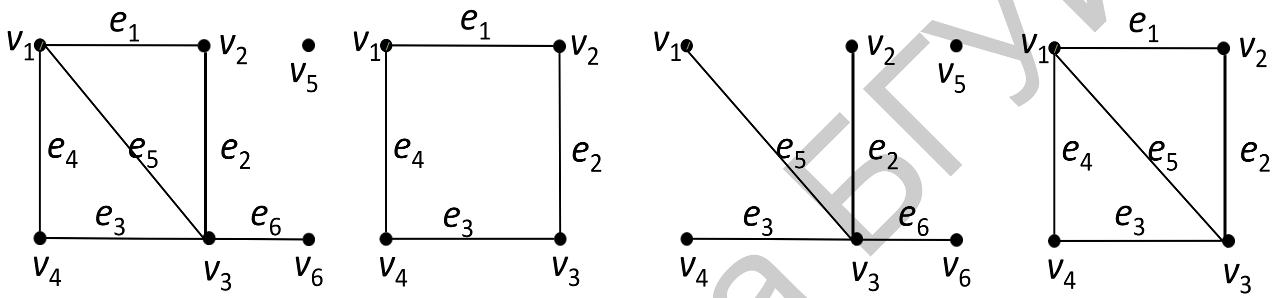
Под «частью графа» понимается некоторый граф, который строится на основе некоторого исходного графа $G = (V, E)$ и содержит некоторое подмножество его вершин и ребер. Рассмотрим некоторые варианты выделения таких частей.

Граф $H = (W, F)$ называется *подграфом* графа $G = (V, E)$, если $W \subseteq V$, $F \subseteq E$ (при этом вершины, инцидентные любому ребру из F, должны принадлежать множеству W). Если $H = (W, F)$ является подграфом графа $G = (V, E)$, то говорят, что он содержится в G (рисунок 4.8).

Выделяются также два частных случая подграфов. Подграф $H = (W, F)$ графа $G = (V, E)$ называется *остовным*, если $W = V$, а $F \subseteq E$. Таким образом, остовный подграф имеет то же множество вершин, что и граф G, но множество ребер является подмножеством множества ребер исходного графа.

Подграф $H = (W, F)$ называется *подграфом* графа $G = (V, E)$, *порожденным множеством* вершин $W \subseteq V$ вершин, если F является множеством всех тех ребер графа G, концы которых содержатся в множестве W.

Примеры приведенных подграфов приведены на рисунке 4.8.



а *б* *в* *г*

а – граф $G = (V, E)$; *б* – подграф $H = (\{v_1,v_2,v_3,v_4\},\{e_1,e_3,e_4\})$;
в – остовный подграф $H = (V, \{e_2,e_3,e_5,e_6\})$;
г – подграф $H = (W, \{e_1,e_2,e_3,e_4,e_5\})$, порожденный множеством $W = \{v_1,v_2,v_3,v_4\}$

Рисунок 4.8 – Примеры подграфов графа G

4.1.6 Ориентированный граф

Ориентированным графом (*орграфом*) называется пара конечных множеств $G = (V, A)$, где V – непустое множество вершин, A – множество упорядоченных пар (v_i, v_j) элементов $v_i, v_j \in V$, называемых *дугами*. Если $a = (v_i, v_j)$ – дуга, то вершины v_i и v_j называются ее концевыми вершинами, причем v_i является *началом*, а v_j – *концом*. Говорят, что дуга *выходит* из начала и *входит* в конец. На графическом представлении графа дуга изображается направленной линией, идущей от начала к концу дуги. В ориентированном графе на рисунке 4.9 началом дуги a_1 является вершина v_1 и концом – вершина v_2.

Основные определения (отношения на множестве вершин и ребер, представления, части и виды графов), приведенные выше для неориентированных графов, легко обобщаются на случай ориентированных графов.

Между вершинами орграфа, так же как и неориентированного, определено отношение смежности, а между вершинами и дугами – отношение *инцидентности*. Две вершины орграфа называются *смежными*, если они являются концевыми для некоторой дуги. Вершина $v \in V$ и дуга $a \in A$ *инцидентны*, если v является началом или концом дуги a.

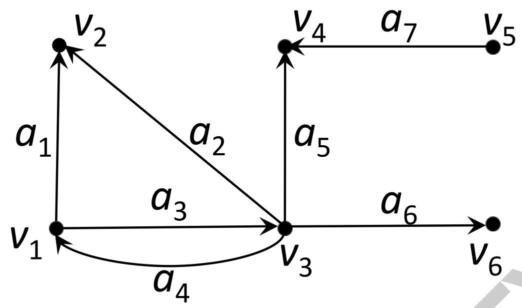


Рисунок 4.9 – Ориентированный граф

Множество всех вершин графа $G = (V, A)$, смежных с вершиной v, называется *окрестностью* вершины v и обозначается символом Γv. Мощность множества Γv, обозначаемая $d(v)$, называется *степенью* вершины v. При этом окрестность вершины Γv делится на две полуокрестности: *полуокрестность исхода* $\Gamma^{+}v$ – множество вершин, в которые входят дуги, исходящие из вершины v, и *полуокрестность захода* $\Gamma^{-}v$ – множество вершин, из которых исходят дуги, заходящие в v. Соответственно мощность множества $\Gamma^{+}v$ называется *полустепенью исхода* и обозначается $d^{+}(v)$, а мощность множества $\Gamma^{-}v$ – *полустепенью захода* и обозначается $d^{-}(v)$. При этом

$$\Gamma v = \Gamma^{+}v \cup \Gamma^{-}v,$$

$$d(v) = d^{+}(v) + d^{-}(v).$$

Для ориентированного графа с множеством дуг A справедливо следующее утверждение (аналог леммы о рукопожатиях для неориентированных графов):

$$\sum_{v \in V} d^{+}(v) = \sum_{v \in V} d^{-}(v) = |A|,$$

т. е. сумма полустепеней исхода всех вершин равна сумме полустепеней захода и равна числу дуг.

Например, для графа, представленного на рисунке 4.9, окрестностью вершины v_3 является $\Gamma v_3 = \{v_1, v_2, v_4, v_6\}$, при этом полуокрестности исхода и захода $\Gamma^{+}v_3 = \{v_1, v_2, v_4, v_6\}$, $\Gamma^{-}v_3 = \{v_1\}$). Соответственно полустепени исхода и захода $d^{+}(v_3) = 4$ и $d^{-}(v_3) = 1$.

Вершину орграфа называют *истоком*, если она имеет нулевую полустепень захода ($d^-(v) = 0$), или *стоком*, если нулю равна полустепень исхода ($d^+(v) = 0$). Орграф с одним истоком и стоком называется *сетью*. На рисунке 4.9 вершина v_5 является истоком, а вершины v_2, v_4, v_6 – стоками.

Для задания ориентированных графов, так же как и неориентированных, используются матрицы смежности и инцидентности.

1. *Матрица смежности* орграфа, так же как и неориентированного, является квадратной булевой матрицей, ее строкам и столбцам соответствуют вершины графа, а элемент на пересечении строки v_i и столбца v_j имеет значение 1 тогда и только тогда, когда в данном графе имеется дуга с началом в вершине v_i и концом в вершине v_j. Граф, показанный на рисунке 4.1, *б*, имеет следующую матрицу смежности:

$$\begin{array}{cccccc|c} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & \\ 0 & 1 & 1 & 0 & 0 & 0 & v_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & v_2 \\ 1 & 1 & 0 & 1 & 0 & 1 & v_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & v_4 \\ 0 & 0 & 0 & 1 & 0 & 0 & v_5 \\ 0 & 0 & 0 & 0 & 0 & 0 & v_6 \end{array}$$

Нетрудно заметить, что матрица смежности орграфа в общем случае не симметрична относительно главной диагонали. Любая строка матрицы смежности орграфа является векторным представлением полуокрестности исхода, а любой столбец – векторным представлением полуокрестности захода соответствующей вершины. Число единиц в i-й строке матрицы смежности равно полустепени исхода i-й вершины графа, а число единиц в j-м столбце равно полустепени захода j-й вершины.

2. *Матрица инцидентности* задает отношение *инцидентн*ости между вершинами и дугами орграфа. Ее строки соответствуют вершинам графа, столбцы – дугам. Элемент на пересечении строки v и столбца a имеет значение 1, если вершина v является началом дуги a, и значение –1, если v является концом дуги a. Если вершина v и дуга a не инцидентны, то указанный элемент имеет значение 0. Матрица инцидентности графа на рисунке 4.1, *б* имеет следующий вид:

a_1	a_2	a_3	a_4	a_5	a_6	a_7	
1	0	1	–1	0	0	0	v_1
–1	–1	0	0	0	0	0	v_2
0	1	–0	1	1	1	0	v_3
0	0	0	0	–1	0	–1	v_4
0	0	0	0	0	0	1	v_5
0	0	0	0	0	–1	0	v_6

Заметим, что любой столбец матрицы инцидентности содержит ровно два элемента, отличных от 0 и сумма их значений равна нулю. Количество единичных элементов любой строки v матрицы инцидентности равно $d^+(v)$, количество элементов со значением «–1» равно $d^-(v)$.

Если граф $G = (V, A)$ является разреженным, то более эффективным, по сравнению с матричным заданием, является его представление посредством списка дуг графа. Например, граф на рисунке 4.9 задается списком $\{(v_1,v_2), (v_3,v_2), (v_1,v_3), (v_3,v_1), (v_3,v_4), (v_3,v_6), (v_5,v_4)\}$.

Для некоторых приложений теории графов важно знать свойства неориентированного графа, получающегося из орграфа $G = (V, A)$ снятием ориентации с его дуг. Неориентированный мультиграф, получающийся в результате снятия ориентации с дуг орграфа, называется его *основанием*. Элементы матрицы смежности основания орграфа $G = (V, A)$ получаются путем сложения элементов матрицы смежности орграфа, симметричных относительно главной диагонали. Например, на рисунке 4.10 приведен орграф и его основание. Соответствующие матрицы смежности имеют вид

v_1	v_2	v_3	v_4	v_5
0	1	0	0	0
0	0	0	1	1
0	1	0	0	0
0	0	0	0	1
0	0	0	1	0

v_1	v_2	v_3	v_4	v_5	
0	1	0	0	0	v_1
1	0	1	1	1	v_2
0	1	0	0	0	v_3
0	1	0	0	2	v_4
0	1	0	2	0	v_5

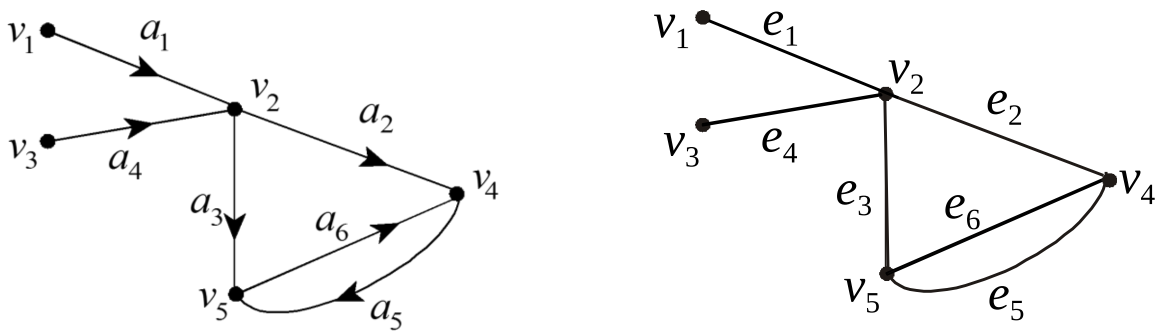


Рисунок 4.10 – Орграф и его основание

4.1.7 Графы и бинарные отношения

Существует полное соответствие между графами и бинарными отношениями. Граф, по сути, представляет собой бинарное отношение на множестве объектов, называемых в данном случае вершинами, кроме того, большим плюсом графов как бинарных отношений является его графическое представление.

Любой неориентированный граф $G = (V, E)$ без кратных ребер задает бинарное отношение, обладающее свойствами симметричности: если ребро $(v, u) \in E$, то пары (v, u) и (u, v) принадлежат отношению $E \subseteq V \times V$. Граф без петель представляет иррефлексивное отношение.

Орграф $G = (V, A)$ без кратных дуг задает бинарное отношение общего вида: если дуга $(v, u) \in A$, то пара (v, u) принадлежит отношению $A \subseteq V \times V$. Изменение направленности дуг орграфа приводит к отношению, обратному исходному, переход к основанию орграфа – к симметричному отношению. Понятие дополнения графа $G = (V, A)$ совпадает с понятием обратного бинарного отношения.

Специальные виды графов задают частные типы отношений. Например, полный неориентированный граф с петлями $G = (V, E)$ задает отношение $E = V \times V$. Операции над графами соответствуют операциям над отношениями.

4.2 Изоморфизм графов

4.2.1 Отношение изоморфизма

Графы, которые получаются один из другого изменением нумерации вершин, называются *изоморфными* друг другу.

Более строго изоморфизм графов можно определить следующим образом. Два графа $G = (V, E)$ и $H = (W, F)$ *изоморфны*, если между множествами их вершин существует взаимно однозначное соответствие, сохраняющее отношение смежности. Другими словами, графы изоморфны, если существует такая биекция $\varphi : V \leftrightarrow W$, что для любых вершин v_i, $v_j \in V$ их образы $\varphi(v_i)$ и $\varphi(v_j)$ смежны в H, если и только если они смежны G.

Графы, изображенные на рисунке 4.11, являются изоморфными, причем

$$\varphi(v_1) = w_2,\ \varphi(v_2) = w_3,\ \varphi(v_3) = w_4,\ \varphi(v_4) = w_1,\ \varphi(v_5) = w_6,\ \varphi(v_6) = w_5.$$

Биекцию $\varphi : V \leftrightarrow W$ можно записать в виде подстановки изоморфизма:

$$\begin{pmatrix} v_1 & v_2 & \dots & v_n \\ \varphi(v_1) & \varphi(v_2) & \dots & \varphi(v_n) \end{pmatrix}$$

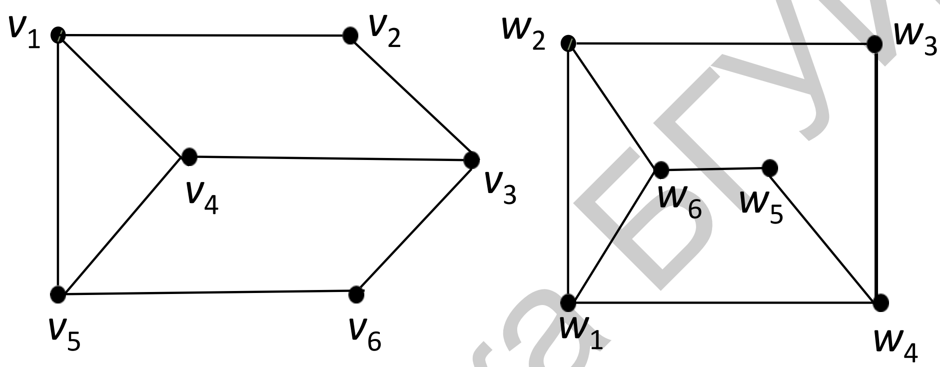


Рисунок 4.11 – Пример изоморфных графов

Например, биекция $\varphi : V \leftrightarrow W$ для графов на рисунке 4.11 задается следующей подстановкой изоморфизма:

$$\begin{pmatrix} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\ w_2 & w_3 & w_4 & w_1 & w_6 & w_5 \end{pmatrix}$$

Изоморфные простые графы можно считать совпадающими (в смысле их графического представления). Как уже говорилось выше, изоморфные графы отличаются только обозначениями вершин, в связи с этим задача установления изоморфизма графов имеет ряд практических приложений, связанных с поиском однотипных объектов, например: при информационном поиске, контроле интегральных схем, определении химических соединений.

4.2.2 Установление изоморфизма графов

Задача установления изоморфизма графов формулируется следующим образом. Заданы два графа, требуется установить, изоморфны они или нет, и если изоморфны, то определить соответствие φ между их вершинами, т. е. найти подстановку изоморфизма.

Очевидно, что необходимым (но не достаточным) условием изоморфизма двух графов является равенство чисел их вершин и равенство чисел ребер, т. е. графы не могут быть изоморфными, если хотя бы одно из этих равенств не выполняется.

Следует отметить сразу алгоритмическую трудность проверки отношения изоморфизма графов в общем случае. Графическое представление графов при решении этой задачи даже для графов довольно скромного размера ничего не дает. Наглядным подтверждением этого служит рисунок 4.12, где приведены три небольших графа, изоморфизм которых далеко не очевиден.

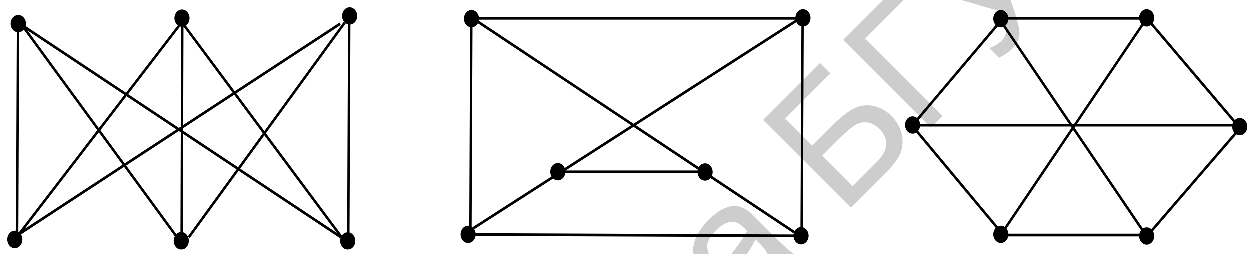


Рисунок 4.12 – Изоморфные графы

Тривиальный способ установления изоморфизма графов основан на анализе их матриц смежности: графы (любого типа) изоморфны тогда и только тогда, когда их матрицы смежности могут быть получены друг из друга перестановкой строк и столбцов. Предполагается, что вершины графов пронумерованы в порядке следования строк и столбцов в этих матрицах, и изменение нумерации вершин графов приводит к перестановке строк и столбцов их матриц смежности. Если зафиксировать матрицу смежности одного графа, а в другом последовательно менять порядок строк (и соответственно столбцов), то в случае изоморфизма на каком-то шаге матрицы совпадут. Если этого не произойдет, то графы, вероятно, неизоморфны, но чтобы убедиться в этом, надо выполнить $n!$ перестановок, где n – число вершин графа.

Отношение изоморфизма графов представляет собой отношение эквивалентности, определенное на множестве графов с одинаковым числом вершин (оно симметрично, транзитивно и рефлексивно). Следовательно, это отношение позволяет произвести разбиение множества графов с числом n вершин на классы эквивалентности. Множество графов, попарно изоморфных друг другу, составляет класс изоморфизма графов. Число классов эквивалентности графов с заданным числом n вершин в зависимости от величины $n = 0, 1, 2, 3,\ldots$ представляет собой быстро возрастающую последовательность (1, 1, 2, 4, 11, 34, 156, 1044, 12346, ...).

Максимальное число ребер в графе с числом n вершин равно $p = n(n-1)/2$, соответственно число всех графов равно 2^p. Например, для $n = 3$ и $n = 4$ всего существует $2^3 = 8$ и $2^6 = 64$ графов, а классов изоморфизма 4 и 11 соответственно. На рисунке 4.13 приведены классы C_1, C_2, C_3 и C_4 изоморфных графов с тремя вершинами.

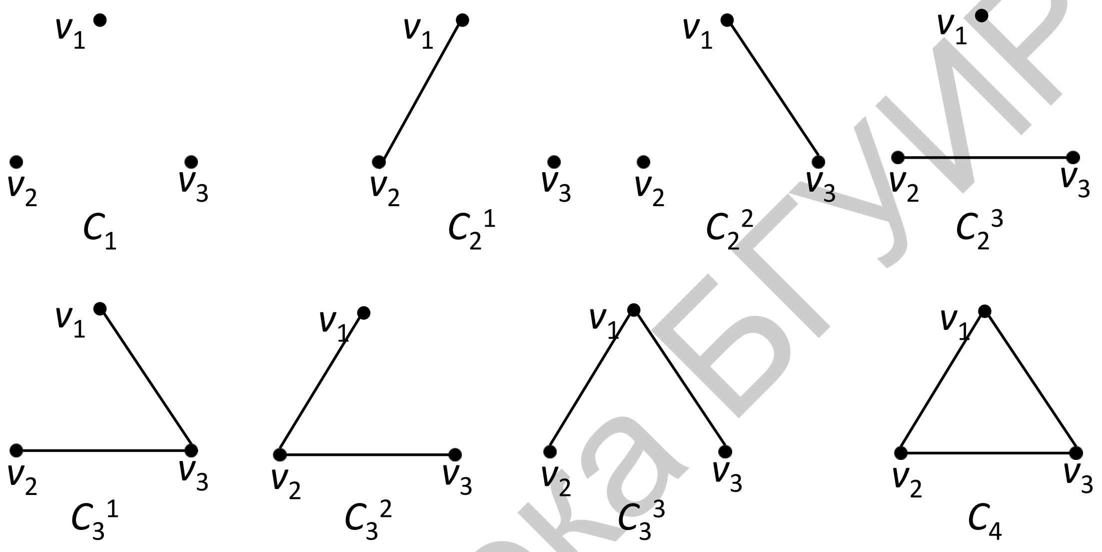


Рисунок 4.13 – Четыре класса изоморфных трехвершинных графов

4.2.3 Канонизация графов

Необходимым (но не достаточным) условием изоморфизма двух графов является равенство числовых характеристик графов, называемых инвариантами, которые совпадают у изоморфных графов.

Инвариантом относительно некоторого преобразования называется величина, не меняющая свое значение при этом преобразовании. Инвариантом графа G является связанное с ним число, которое принимает одно и то же значение для любого графа, изоморфного G. Полный набор инвариантов графа определяет граф с точностью до изоморфизма.

При установлении изоморфизма рассматриваются инварианты графа и инварианты вершин. Инвариантами вершины являются степени (полустепени), число вершин, отстоящих от данной вершины на определенном расстоянии. Инвариантами графа являются, например, число $n(G)$ вершин, число $m(G)$ ребер, число компонент связности, хроматическое число, упорядоченный по возрастанию или убыванию вектор степеней вершин $s(G) = (d(v_1), d(v_1), \ldots, d(v_n))$ и др. Различные инварианты имеют различную трудоемкость вычисления. В настоящее время полный набор инвариантов графа, вычислимый за полиномиальное время, не известен, однако не доказано, что он не существует.

Совокупность инвариантов вершин графа служит его инвариантом, в этом случае эта совокупность должна быть упорядоченной.

Сокращение перебора при решении задачи установления изоморфизма графов основано на приведении сравниваемых графов к каноническому виду. *Канонизация графа* заключается в упорядочении его вершин по значениям их инвариантов. Если для вершин графа имеется система инвариантов $\alpha_1, \alpha_2, \dots, \alpha_p$, то считается, что задано отношение частичного порядка «$\prec$» на множестве вершин графа $V = \{v_1, v_2, \dots, v_n\}$, такое, что $v_i \prec v_j$, если $\alpha_k(v_i) < \alpha_k(v_j)$ для некоторого $k \in \{1, 2, \dots, p\}$ и $\alpha_l(v_i) = \alpha_l(v_j)$ для всех $l < k$.

Полная канонизация графа достигается, когда порядок $\prec$ оказывается полным и строгим. Матрицы смежности полностью канонизированных изоморфных графов должны совпадать. Но даже если оказывается невозможным полностью канонизировать анализируемые на изоморфизм графы, но удается разбить множество вершин V канонизируемого графа на подмножества $V_1, V_2, \dots, V_k$, характеризуемые совпадением инвариантов входящих в них вершин, упомянутый выше перебор при установлении изоморфизма можно значительно сократить, так как он проводится только для данных подмножеств. Действительно, если $|V| = n$ и $|V_i| = n_i$ ($i = 1, 2, \dots, k$), то $n_1!n_2! \dots n_k! << n!$

Рассмотрим один из простых методов канонизации графа. Разобьем множество V вершин графа G сначала на подмножества $V_1, V_2, \dots, V_k$, число k которых равно числу различных степеней вершин и каждое из которых включает вершины с одинаковой степенью.

Для каждой вершины $v_i \in V$ образуем вектор размерности k, компоненты которого соответствуют множествам $V_1, V_2, \dots, V_k$ и значением j-й компоненты является число вершин из множества V_j, смежных с v_i. Если в одном и том же V_i ($i = 1, 2, \dots, k$) окажутся вершины с различными векторами, то разобьем это V_i так, чтобы в каждом из получившихся подмножеств остались вершины с одинаковыми векторами. Соответственно после разбиения хотя бы одного подмножества V_i увеличивается размерность k векторов (соответствующих вершинам) и их компонентам придаются новые значения. При этом поддерживаем порядок следования вершин, соответствующий лексикографическому порядку их векторов. Данное преобразование повторяем до тех пор, пока в любом из $V_1, V_2, \dots, V_k$ не останутся вершины только с одинаковыми векторами размерностью k.

Проиллюстрируем описанный процесс канонизации на примере графов, изображенных на рисунке 4.11. При установлении изоморфизма между двумя графами следует параллельно канонизировать оба графа. Тогда, если встретится какое-нибудь несовпадение, можно прекращать процесс и выносить решение об отсутствии изоморфизма. В качестве начального инварианта вершины возьмем ее степень, т. е. $\alpha(v) = d(v)$. Результаты выполнения описанного процесса канонизации по шагам выглядят для графов $G = (V, E)$ и $H = (W, F)$ следующим образом:

		α
V_1	v_2	2
	v_6	2
	v_1	3
V_2	v_3	3
	v_4	3
	v_5	3

		α_1	α_2
V_1	v_2	0	2
	v_6	0	2
	v_1	1	2
V_2	v_3	2	1
	v_4	0	3
	v_5	1	2

		α_1	α_2	α_3	α_4
V_1	v_2	0	0	1	1
	v_6	0	0	1	1
V_2	v_4	0	0	2	1
V_3	v_1	1	1	1	0
	v_5	1	1	1	0
V_4	v_3	2	1	0	0

		α
W_1	w_3	2
	w_5	2
	w_1	3
W_2	w_2	3
	w_4	3
	w_6	3

		α_1	α_2
W_1	w_3	0	2
	w_5	0	2
	w_1	0	3
W_2	w_2	1	2
	w_4	2	1
	w_6	1	2

		α_1	α_2	α_3	α_4
W_1	w_3	0	0	1	1
	w_5	0	0	1	1
W_2	w_1	0	0	2	1
W_3	w_2	1	1	1	0
	w_6	1	1	1	0
W_4	w_4	2	1	0	0

В полученных в результате канонизации разбиениях множеств вершин подмножества V_1 и V_3 (и W_1 и W_3) имеют по две вершины, это значит, что при установлении изоморфизма требуется дополнительный анализ – перебор перестановок на множестве вершин этих подмножеств. Так как каждое из подмножеств содержит по две вершины, то необходимо перебрать четыре варианта упорядочения вершин, например, графа $H = (W, F)$ (при фиксированном упорядочении вершин графа $G = (V, E)$ в подмножества V_1 и V_3):

1) $(w_3, w_5), (w_2, w_6)$;

2) $(w_3, w_5), (w_6, w_2)$;

3) $(w_5, w_3), (w_6, w_2)$;

4) $(w_5, w_3), (w_2, w_6)$

и сравнить матрицы смежности графов $G = (V, E)$ и $H = (W, F)$ при этих упорядочениях.

Матрицы смежности графов $G = (V, E)$ и $H = (W, F)$ после перенумерации вершин в процессе их канонизации имеют следующий вид:

v_2	v_6	v_4	v_1	v_5	v_3		w_3	w_5	w_1	w_2	w_6	w_4
0	0	0	1	0	1		0	0	0	1	0	1
0	0	0	0	1	1		0	0	0	0	1	1
0	0	0	1	1	1		0	0	0	1	1	1
1	0	1	0	1	0		1	0	1	0	1	0
0	1	1	1	0	0		0	1	1	1	0	0
1	1	1	0	0	0		1	1	1	0	0	0

Матрицы равны, следовательно, графы изоморфны, соответствующую биекцию $\varphi : V \leftrightarrow W$ можно записать в виде следующей подстановки изоморфизма:

$$\begin{pmatrix} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\ w_2 & w_3 & w_4 & w_1 & w_6 & w_5 \end{pmatrix}$$

Следует отметить, что при установлении изоморфизма графов используемая выше операция канонизации не всегда позволяет сократить перебор перестановок на множестве их вершин. Например, канонизация однородных графов (рисунок 4.14) бесполезна.

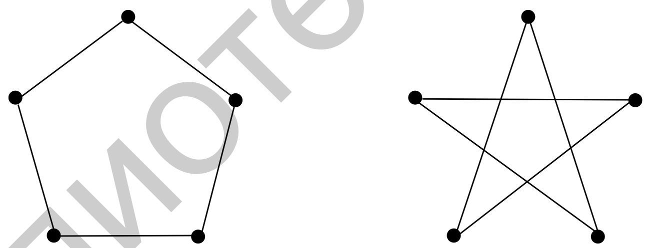


Рисунок 4.14 – Изоморфные однородные графы

Можно заметить, что графы на рисунке 4.14 являются дополнениями друг друга. Граф, изоморфный своему дополнению, называется *самодополнительным*.

4.3 Обходы графа

Решение многих задач на графах основывается на обходе графа, который представляет собой некоторый систематический просмотр его вершин или ребер. Такой обход можно выполнить многими способами, но наибольшее распространение получили две стратегии обхода – *поиск в ширину* и *поиск в глубину*. С помощью поиска в ширину можно решить, например, такие задачи теории графов, как нахождение компонент связности графа, множеств вершин, достижимых из заданной, кратчайших цепей из одной заданной вершины в другую. Поиск в глубину лежит в основе многих алгоритмов теории графов, например рассматриваемых ниже алгоритмов поиска эйлеровых и гамильтоновых циклов.

При поиске в ширину, начиная из заданной начальной вершины v связного графа, которую помечаем как вершину уровня 0, все вершины из Γv помечаем как вершины первого уровня. Затем все непомеченные вершины из окрестностей вершин первого уровня помечаем как вершины второго уровня и т. д., непомеченные вершины из окрестностей вершин уровня k помечаем как вершины уровня $k + 1$. Просмотр вершин прекращается, когда все вершины будут помечены.

В процессе поиска в глубину каждой пройденной вершине v связного графа приписывается номер $N(v)$ (соответствующий глубине просмотра), а проходимые ребра помечаются. Сначала, начиная из заданной вершины v_0 графа, которой приписывается $N(v_0) = 1$, строится цепь путем выбора на каждом k-м шаге непомеченного ребра и вершины, не имеющей номера, из окрестности вершины с $N(v) = k–1$. Пусть процесс дошел до вершины u, которой присвоен номер $N(u)$ и P – последний присвоенный вершинам номер. Возможны следующие ситуации:

– имеется некоторое непомеченное ребро $(u\ w)$ и вершина w имеет номер, тогда помечаем это ребро как обратное и продолжаем поиск непомеченного ребра, инцидентного вершине u;

– имеется некоторое непомеченное ребро $(u\ w)$ и вершина w не имеет номера, присваиваем ей номер $N(w) = P + 1$ и считаем ее получившей номер из u, ребро помечаем как прямое и будем дальше рассматривать вершину w;

– все ребра, инцидентные вершине u, помечены, тогда возвращаемся к вершине, из которой u получила номер.

Процесс обхода графа заканчивается, если все ребра помечены и произошел возврат в вершину v_0.

4.3.1 Достижимость и связность

Любая чередующаяся последовательность вершин $v_i \in V$ и ребер $e_j \in E$ графа $G = (V, E)$ вида

$$v_1, e_1, v_2, e_2, \dots, e_k, v_{k+1},$$

в которой любые соседние элементы являются инцидентными, называется *маршрутом* (или (v_1,v_{k+1})-маршрутом). Это определение годится также и для графов более общего вида: псевдо-, мульти- и орграфа. Маршрут простого графа может быть однозначно задан указанием последовательности его вершин или ребер:

$$v_1, v_2, \ldots, v_{k+1}, \quad e_1, e_2, \ldots, e_k.$$

Маршрут может быть конечным либо бесконечным. Одно и то же ребро может встречаться в маршруте не один раз. *Длиной маршрута* называется количество входящих в него ребер, причем каждое ребро считается столько раз, сколько оно встречается в данном маршруте.

Маршрут, все ребра которого различны, называется *цепью*. Цепь, все вершины которой различны (кроме, может быть, крайних), называется *простой цепью* или *путем*. Маршрут, цепь, путь называются открытыми, если их концевые вершины различны, и замкнутыми (циклическими) – в противном случае. Замкнутая простая цепь $v_1, e_1, v_2, e_2, \ldots, e_k, v_1$ называется *циклом*, а замкнутая простая цепь – *простым циклом*.

Например, последовательность $v_3, e_5, v_2, e_1, v_1, e_3, v_6, e_6, v_3, e_5, v_2$ является маршрутом длиной 5 в графе на рисунке 4.15, *а*, но не является цепью, так как в нем повторяется ребро e_5. Последовательность $v_3, e_5, v_2, e_1, v_1, e_4, v_3, e_6, v_6$ длиной 4 является цепью, но не является простой цепью, так как в ней повторяется вершина v_3. Последовательность $v_3, e_5, v_2, e_1, v_1, e_2, v_4$ длиной 3 является простой цепью, связывающей вершины v_3 и v_4, однако расстояние между этими вершинами равно 2, так как в графе существует цепь v_3, e_6, v_6, e_5, v_4 длиной 2. Циклом является замкнутая цепь $v_3, e_5, v_2, e_1, v_1, e_4, v_3$.

Граф без циклов называется *ациклическим*.

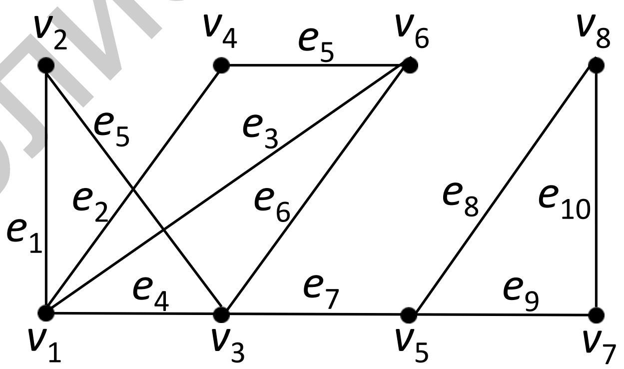


Рисунок 4.15 – Связный неориентированный граф

Любую цепь и любой цикл графа можно рассматривать как его подграф.

Вершина v_i в неориентированном графе является *достижимой* из вершины v_j, если в этом графе имеется путь с началом в v_i и концом в v_j. С понятием длины цепи связано понятие *расстояния* в графе. Под расстоянием $l(v, u)$ между вершинами v и u понимается длина кратчайшей простой цепи, связывающей данные вершины, а сама такая цепь называется *геодезической*. Длина длинней-шей геодезической цепи графа называется его *диаметром*.

Граф называется *связным*, если каждая его вершина достижима из любой другой вершины или, другими словами, между любыми двумя вершинами имеется цепь. Максимальный связный подграф графа (не содержащийся ни в каком другом его связном подграфе) называется *компонентой связности* или просто *компонентой* данного графа. Например, связными являются графы на рисунках 4.14, 4.3, *а*, *б*, *в*. Граф, приведенный на рисунке 4.16, не является связным.

В связном простом графе расстояние $l(v, u)$ является метрикой, так как удовлетворяет следующим свойствам, имеющим место для любых трех вершин $v, u, w \in V$:

1) $l(v, u) \geq 0$ и $l(v, u) = 0$ только для случая $v = u$;
2) $l(v, u) = l(u, v)$;
3) $l(v, u) + l(u, w) \geq l(v, w)$.

Нетрудно заметить, что отношение достижимости на множестве вершин неориентированного графа является отношением эквивалентности на множестве V, так как оно симметрично, транзитивно и его можно считать рефлексивным. Классы эквивалентности $V_1, V_2, \ldots, V_k$, порождаемые отношением достижимости на множестве вершин, определяют порожденные подграфы графа $G = (V, E)$, которые являются его *компонентами связности*.

Например, граф на рисунке 4.16 имеет две компоненты связности, порожденные множествами вершин $V_1 = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ и $V_2 = \{v_7, v_8, v_9, v_{10}\}$.

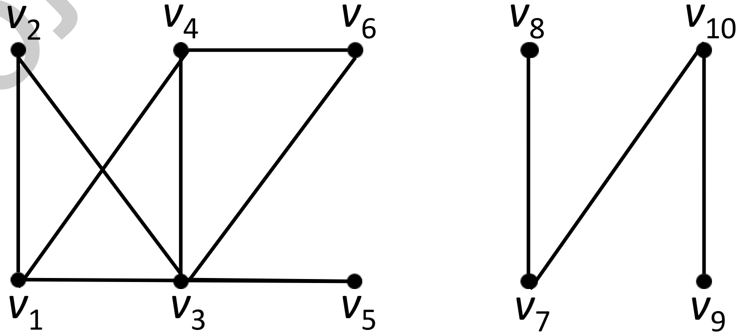


Рисунок 4.16 – Несвязный граф с двумя компонентами связности

С понятием связности связана *мера вершинной* (реберной) *связности* графа, которая определяется минимальным числом вершин (и ребер), удаление которых приводит к несвязному или одновершинному графу. Говорят, что вершина v графа G является *точкой сочленения*, если граф $G - v$, получаемый путем удаления вершины v, имеет больше компонент связности, чем исходный граф. Аналогично ребро e графа G является *мостом*, если граф $G - e$, получаемый путем удаления e, имеет больше компонент связности, чем исходный граф.

Из определения вершинной (реберной) связности следует, что связность несвязного графа равна 0, вершинная (реберная) связность связного графа, имеющего точку сочленения (мост), равна 1. Мера вершинной связности полного графа K_n равна $n - 1$.

К примеру, граф на рисунке 4.15 имеет две точки сочленения – вершины v_3 и v_5 графа и один мост – ребро e_7, соответственно меры вершинной и реберной связности этого графа равны единице.

Для определения связности графа может быть использован поиск в ширину в процессе построения дерева поиска. Если граф является связным, то в процессе поиска в ширину будут пройдены все вершины. При реализации поиска в ширину используется стек S (запись нового элемента производится только в конец этого массива и выборка из него также производится с конца), в который записываются вершины, подлежащие просмотру. Поиск начинается с выбора любой вершины $v \in V$, занесения ее в стек и искомое множество V_1 вершин, достижимых из v. Далее на каждом шаге из стека выбирается вершина u (последняя из занесенных в него), она становится текущей. Для u находится множество вершин $V_u = \Gamma u \setminus V_1$ тех вершин из ее окрестности, которые ранее не рассматривались, вершины из V_u записываются в стек и в множество V_1. Процесс формирования множества достижимости заканчивается, когда стек опустеет. Если полученное множество $V_1 = V$, граф связен, иначе он несвязен и найдена одна из компонент связности. В последнем случае можно найти следующую компоненту связности на множестве вершин $V \setminus V_1$.

Проиллюстрируем работу алгоритма на примере графа, приведенного на рисунке 4.16.

$S = \{v_1\}$, $V_1 = \{v_1\}$;

1) v_1, $V_{v1} = \Gamma v_1 = \{v_2, v_3, v_4\}$, $S = \{v_2, v_3, v_4\}$, $V_1 = \{v_1, v_2, v_3, v_4\}$;

2) v_4, $V_{v4} = \{v_6\}$, $S = \{v_2, v_3, v_6\}$, $V_1 = \{v_1, v_2, v_3, v_4, v_6\}$;

3) v_6, $V_{v4} = \{\ \}$, $S = \{v_2, v_3\}$, $V_1 = \{v_1, v_2, v_3, v_4, v_6\}$;

4) v_3, $V_{v3} = \{v_5\}$, $S = \{v_2, v_5\}$, $V_1 = \{v_1, v_2, v_3, v_4, v_5, v_6\}$;

5) v_5, $V_{v5} = \{\ \}$, $S = \{v_2\}$, $V_1 = \{v_1, v_2, v_3, v_4, v_5, v_6\}$;

6) v_2, $V_{v2} = \{\ \}$, $S = \{\ \}$, $V_1 = \{v_1, v_2, v_3, v_4, v_5, v_6\}$.

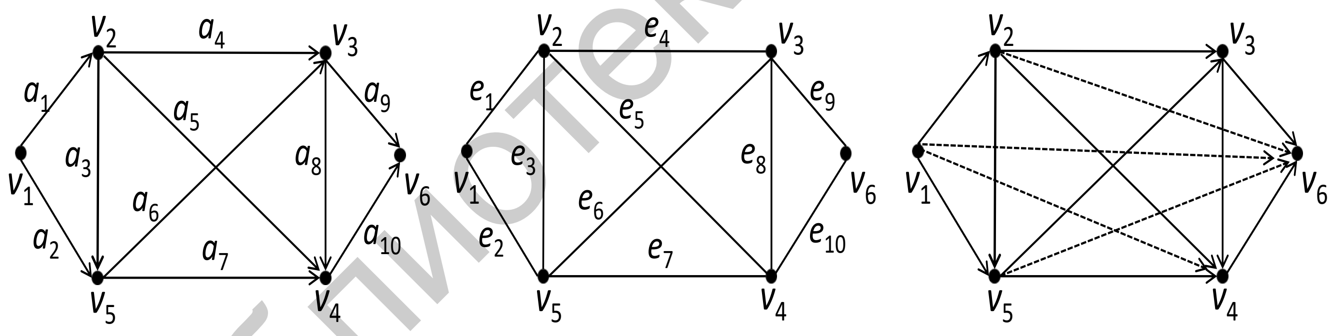
В результате делаем вывод, что анализируемый граф не является связным, так как $V \setminus V_1 = \{v_7, v_8, v_9, v_{10}\} \neq \varnothing$ и найдена первая компонента связности. Выполняя алгоритм на множестве вершин $V' = \{v_7, v_8, v_9, v_{10}\}$, можно аналогично найти вторую компоненту связности.

Определения *маршрута*, *цепи* и *пути* для орграфа аналогичны вышеприведенным определениям для неориентированного графа. Маршрут, цепь и путь для орграфа называют *ориентированными*. В ориентированном маршруте дуги и вершины могут повторяться. Если начальная и конечная вершины ориентированного маршрута, цепи или пути совпадают, то они называются замкнутыми, иначе открытыми. Замкнутый ориентированный путь $v_1, a_1, v_2, a_2, \ldots, a_k, v_1$ называется *ориентированным циклом* или *контуром*.

Аналогично вершина v_j в ориентированном графе является *достижимой* из вершины v_j, если в этом графе имеется путь с началом в v_i и концом в v_j.

Например, вершина v_3 (рисунок 4.17, *а*) достижима из вершины v_1, так как существует путь v_1, a_1, v_2, a_4, v_3, но не достижима из вершины v_4.

В орграфе отношение связанности между вершинами может быть несимметричным, поэтому определение связности для орграфов существенно отличается от аналогичного определения для неориентированного графа.



а *б* *в*

Рисунок 4.17 – Ориентированный граф (*а*), его основание (*б*) и транзитивное замыкание (*в*)

Две вершины v и u ориентированного графа называются:

– *сильно связными*, если существуют ориентированные цепи из v в u и из u в v;

– *односторонне связными*, если существует ориентированная цепь либо из v в u, либо из u в v;

– *слабо связными*, если они связаны в основании орграфа (рисунок 4.17, *б*), полученного отменой ориентации дуг.

Например, в орграфе на рисунке 4.17, *а* нет ни одной пары сильно связных вершин, но все пары вершин слабо связны, например, v_1 и v_3, v_1 и v_4, v_2 и v_6.

Орграф называется:

– *сильно связным*, если все пары его вершин являются сильно связными;

– *односторонне связным*, если все пары его вершин являются односторонне связными;

– *слабо связным*, если все пары его вершин являются слабо связными.

Из сильной связности орграфа следует его односторонняя и слабая связность, а из односторонней связности следует слабая связность.

Например, граф на рисунке 4.17, *а* не является сильно связным, но он будет односторонне связным.

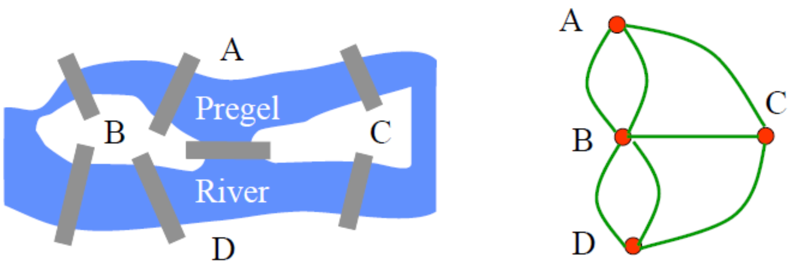
Ориентированный граф называется *транзитивным*, если из существования дуг $a_p = (v_i,v_j)$ и $a_q = (v_j,v_k)$ следует существование дуги $a_r = (v_i,v_k)$. *Транзитивным замыканием* ориентированного графа $G = (V, A)$ называется граф $G^* = (V, A^*)$, где A^* получено из A добавлением минимально возможного количества дуг, необходимого для того, чтобы граф G^* был транзитивным. На рисунке 4.17, *в* приведен граф G^*, являющийся транзитивным замыканием орграфа G (см. рисунок 4.17, *а*). Добавленные при этом дуги показаны пунктирными линиями.

4.3.2 Эйлеровы графы

Началом теории графов считается работа Л. Эйлера, опубликованная в 1736 г., в которой он решил *задачу о кенигсбергских мостах*. На протекающей через город Кенигсберг (в настоящее время г. Калининград) реке Прегель расположены два острова, которые были соединены семью мостами между собой и с берегами реки (рисунок 4.18, *а*). Задача заключалась в том, чтобы найти такую точку суши, выйдя из которой, можно пройти по всем мостам по одному разу и вернуться в нее обратно. Эйлер привел формальное решение этой задачи.

На языке теории графов эта задача формулируется следующим образом: в заданном связном графе (мультиграфе) необходимо выделить цикл, содержащий все ребра этого графа. В исходной задаче вершинам мультиграфа соответствуют участки суши A, B, C и D, а ребрам – мосты через реку (см. рисунок 4.18, *б*). Эйлер нашел условия существования такого цикла.

Цикл, содержащий все ребра графа, носит название *эйлерова цикла*. Цепь с этим же свойством называется *эйлеровой цепью*. Очевидно, что в эйлеровы цикл и цепь входят не только все ребра графа, но и вершины, причем последние могут и повторяться.



а *б*

а – план; *б* – мультиграф, соответствующий плану

Рисунок 4.18 – Кенигсбергские мосты

Теорема Эйлера. Связный неориентированный мультиграф имеет эйлеров цикл тогда и только тогда, когда степени всех его вершин четны. В связном неориентированном графе существует эйлерова цепь тогда и только тогда, когда он имеет не более двух вершин с нечетной степенью.

Связный граф, имеющий эйлеров цикл, называется *эйлеровым графом*. Если связный граф не содержит эйлеров цикл, но содержит эйлерову цепь, то он называется *полуэйлеровым графом* (рисунок 4.19).

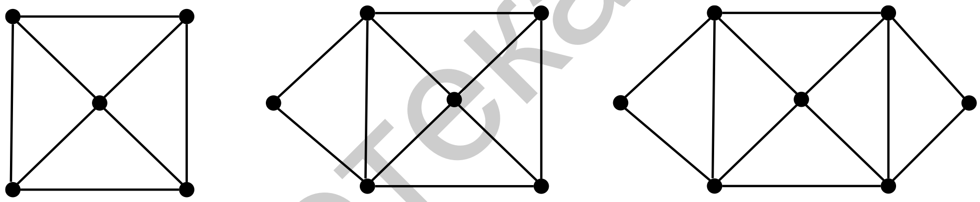


Рисунок 4.19 – Графы: неэйлеров; полуэйлеров и эйлеров

Эйлеров цикл в связном мультиграфе, если известно, что он существует, может быть получен с помощью *алгоритма Флери*, который состоит в последовательном обходе и удалении пройденных ребер графа. При обходе должны соблюдаться следующие правила.

1. Выходим из произвольной вершины и удаляем каждое пройденное ребро из графа, помещая его в формируемую цепь.

2. Отправляясь из очередной вершины, выбираем для прохода любое инцидентное ей ребро, причем мост выбираем только в том случае, когда нет другой возможности.

3. Алгоритм заканчивает работу, когда все ребра удалены, а цикл (начало и конец цепи совпадут, так как все вершины имеют четную степень) сформирован.

Трудоемкость этого алгоритма, с учетом того что на каждом шаге необходимо проверять, является ли ребро мостом, оценивается как $O(m^2)$, где m – число ребер графа. На рисунке 4.20 продемонстрирована работа алгоритма Флери: начиная с вершины *v*, ребра пронумерованы в порядке их прохождения.

Если связный граф имеет две вершины с нечетной степенью, то в нем можно найти только эйлерову цепь, начиная ее поиск с одной из этих двух вершин и заканчивая в другой. При этом следует выполнять те же правила, что и при поиске эйлерова цикла.

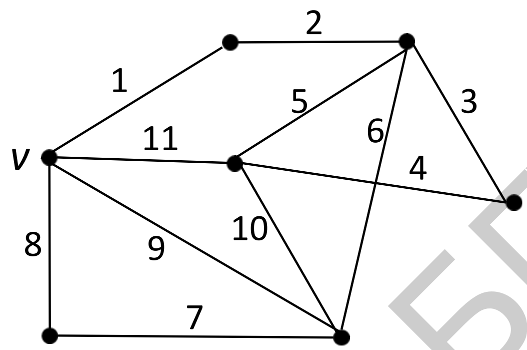


Рисунок 4.20 – Построение эйлерова цикла

Следует заметить, что доля графов, которые являются эйлеровыми, крайне мала.

Задача о нахождении эйлеровых циклов и цепей встречается в различных головоломках и занимательных задачах. Например, требуется нарисовать фигуру, называемую саблями (или знаками) Магомета, не отрывая карандаша от бумаги и не повторяя линий. На рисунке 4.21 для иллюстрации решения этой задачи приведены вершины графа, к поиску циклов в котором она сводится. К поиску эйлерова цикла сводится задача обхода выставки по различным коридорам (залам), в которых выставлены картины и которые необходимо пройти, чтобы посмотреть все экспонаты выставки ровно один раз.

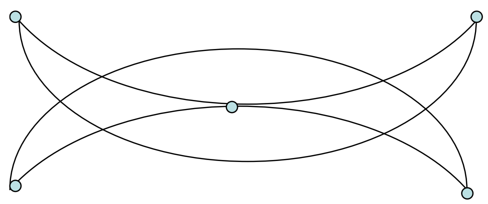


Рисунок 4.21 – Сабли Магомета

Обобщением задачи Эйлера является известная *задача китайского почтальона*, которая имеет разнообразные приложения (например, при проверке электрических сетей) и ставится следующим образом. Имеется взвешенный G, каждому ребру e_i которого приписывается положительный вес $c(e_i)$ (длина). Требуется найти замкнутый маршрут, проходящий через каждое ребро графа G по крайней мере один раз и такой, что сумма величин $n_i c(e_i)$, где n_i – число прохождений ребра e_i, минимальна. Если граф является эйлеровым, то любой такой маршрут представляет собой эйлеров цикл, а данная сумма одинакова для всех эйлеровых циклов и является суммой весов всех ребер.

Случай орграфов. Для того чтобы связный орграф имел эйлеров цикл (рисунок 4.22), необходимо и достаточно, чтобы полустепени исхода и захода всех его вершин были равны ($d^+(v) = d^-(v)$ для любой вершины).

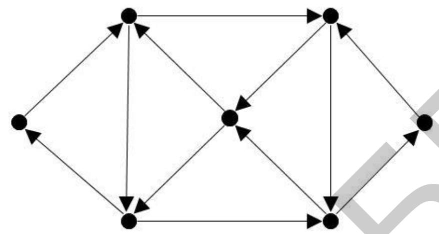


Рисунок 4.22 – Орграф с эйлеровым циклом

Для того чтобы связный орграф имел эйлерову цепь, необходимо и достаточно, чтобы: 1) полустепени исхода и захода всех его вершин, кроме двух v_i и v_j, были равны; 2) $d^+(v_i) = d^-(v_i) + 1$, $d^+(v_j) = d^-(v_j) - 1$.

4.3.3 Гамильтоновы графы

Понятие *гамильтонова* графа связывают с именем ирландского математика В. Гамильтона, который предложил в 1859 г. игру «Кругосветное путешествие». В этой игре каждой из 20 вершин додекаэдра приписано название одного из городов мира. Требовалось, переходя от одного города к другому по ребрам додекаэдра посетить каждый город в точности один раз и вернуться в исходную точку.

Напомним, что додекаэдр – правильный многогранник, составленный из двенадцати правильных пятиугольников, являющихся его гранями, каждая вершина додекаэдра является вершиной трех правильных пятиугольников, а сам додекаэдр имеет 12 пятиугольных граней, 30 ребер и 20 вершин (в каждой сходятся 3 ребра).

Задача обхода вершин додекаэдра сводится к задаче обхода вершин соответствующего ему плоского графа (рисунок 4.23), построенного на ребрах додекаэдра. Задача поиска пути на додекаэдре таким образом была сведена к поиску простого цикла графа, проходящего через каждую вершину графа ровно по одному разу. Существование такого цикла равносильно существованию циклической последовательности ходов, содержащей каждую позицию по разу.

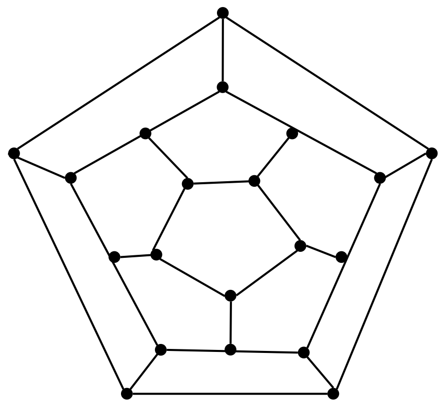


Рисунок 4.23 – Плоский граф, соответствующий додекаэдру

Цикл называется *гамильтоновым*, если он проходит через каждую вершину графа ровно один раз. Гамильтонов цикл является остовным простым циклом графа. *Гамильтоновой цепью* называется простая цепь, проходящая каждую вершину графа ровно один раз. Граф, содержащий гамильтонов цикл, называется *гамильтоновым графом*. Граф, содержащий гамильтонову цепь, называется *полугамильтоновым*. Следует заметить, что всякий гамильтонов граф является и полугамильтоновым.

На рисунке 4.24 приведены примеры гамильтонова графа с выделенным циклом (v_1, v_2, v_5, v_4, v_3, v_1) и полугамильтонова графа с выделенной цепью (v_1, v_2, v_5, v_3, v_4).

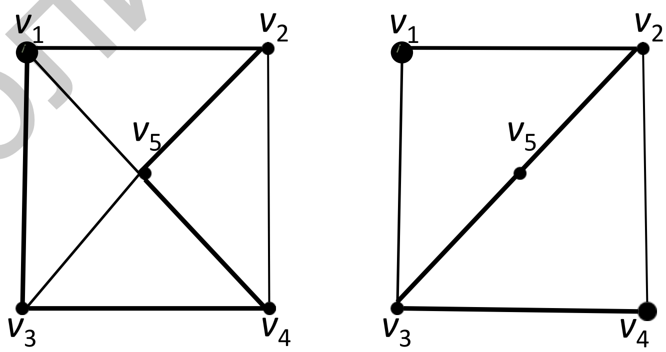


Рисунок 4.24 – Графы: гамильтонов и полугамильтонов

К поиску гамильтоновых циклов сводится много занимательных задач, например, следующие две из них:

1) задача о шахматном коне, которая формулируется следующим образом: можно ли, начиная с произвольного поля шахматной доски, пройти конем в такой последовательности, чтобы посетить каждое из 64 полей и вернуться в исходное;

2) задача про банкет, дающая ответ на вопрос, можно ли компанию из нескольких человек рассадить за круглым столом так, чтобы любые два человека, сидящие рядом, были знакомы.

На первый взгляд задача нахождения гамильтонова цикла в графе похожа на задачу поиска эйлерова цикла. На самом же деле эти задачи принципиально различны. Для произвольного графа до сих пор не известны необходимые и достаточные условия существования гамильтоновых циклов (и цепей), а все известные алгоритмы их поиска требуют перебора большого числа вариантов (кроме графов некоторых специальных видов). Очевидными простейшими необходимыми условиями существования гамильтонова цикла или цепи являются связность графа и отсутствие в нем точек сочленения (вершин, удаление которых увеличивает число компонент связности графа). Простейшим достаточным условием гамильтоновости графа является полнота графа, ибо всякий полный граф K_n (при любом $n \geq 3$) имеет гамильтонов цикл. Гамильтоновым является также полный двудольный граф $K_{k,k}$, т. е. граф с равномощными долями.

Поиск гамильтонова цикла или гамильтоновой цепи в произвольном графе значительно более трудоемкий, чем поиск эйлерова цикла или эйлеровой цепи. Если в n-вершинном графе фиксировать одну вершину и всегда начинать обход именно с нее, то всякому гамильтоновому циклу очевидным образом будет соответствовать перестановка остальных $n - 1$ вершин. Таким образом, чтобы убедиться в негамильтоновости графа потребуется в худшем случае $(n - 1)!$ перестановок. Столько же перестановок потребуется и при поиске гамильтоновой цепи. Следует заметить, что среди всех графов доля графов, являющихся гамильтоновыми, велика.

Рассмотрим один из способов построения гамильтонова цикла в графе, основанный на последовательном построении всех циклов графа методом поиска в глубину. Для построения будем использовать дерево поиска с корнем – началом перебора. Пусть вершины заданного графа $G = (V, E)$ пронумерованы в произвольном порядке: $v_1, v_2, \ldots, v_n$. Для каждой вершины v_i сформируем ее окрестность Γv_i. Будем представлять цикл в виде последовательности C вершин. В качестве отправной возьмем первую в порядке нумерации вершину v_1, объявим ее первым элементом получаемой последовательности C. К вершине v_1 припишем вершину v_j, первую в списке Γv_1, в результате чего получим цепь $C = (v_1, v_j)$. Аналогично из списка Γv_j вновь включенной в цепь вершины v_j, выберем первую вершину v_k, не присутствующую в цепи C, получим цепь $C = (v_1, v_j, v_k)$ и т. д.

Пусть получена последовательность $C = (v_1, v_j, v_k, \dots , v_q, v_r)$, не содержащая все вершины из V, но в списке Γv_r нет вершин, не содержащихся в C. Тогда делаем шаг назад и в списке Γv_q выбираем вместо вершины v_r следующую по порядку вершину. Если в Γv_q такой вершины нет, делаем еще шаг назад и обращаемся к окрестности предшествующей вершины и т. д. В результате либо получаем искомую последовательность, когда все вершины из V вошли в C, либо, возвращаясь к списку Γv_1, обнаруживаем, что он оказывается исчерпанным. В последнем случае граф не имеет гамильтонова цикла.

Продемонстрируем описанный процесс на примере графа, приведенного на рисунке 4.25.

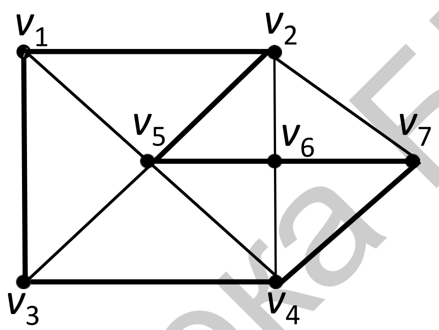


Рисунок 4.25 – Граф с выделенным гамильтоновым циклом

Зададим данный граф перечислением списков окрестностей его вершин:

$$\Gamma v_1 = \{v_2, v_3, v_5\};$$
$$\Gamma v_2 = \{v_1, v_5, v_6, v_7\};$$
$$\Gamma v_3 = \{v_1, v_4, v_5\};$$
$$\Gamma v_4 = \{v_3, v_5, v_6, v_7\};$$
$$\Gamma v_5 = \{v_1, v_2, v_3, v_4, v_6\};$$
$$\Gamma v_6 = \{v_2, v_4, v_5, v_7\};$$
$$\Gamma v_7 = \{v_2, v_4, v_6\}.$$

Сначала получаем последовательность $C = (v_1, v_2, v_5, v_3, v_4, v_7, v_6)$ и в списке Γv_6 нет вершин, не присутствующих в C. Шаг назад приводит к последовательности $C = (v_1, v_2, v_5, v_3, v_4, v_7)$. В списке вершины Γv_7 также нет нерассмотренных вершин, не присутствующих в последовательности C. Возвращаемся к последовательностям с тупиковыми вершинами $C = (v_1, v_2, v_5, v_3, v_4)$, $C = (v_1, v_2, v_5, v_3)$ и $C = (v_1, v_2, v_5)$. Из $\Gamma v_5 =\{v_4, v_6\}$ выбираем v_4 и получаем последовательность $C = (v_1, v_2, v_5, v_4)$. Затем получаем последовательности

$C = (v_1, v_2, v_5, v_4, v_3)$, $C = (v_1, v_2, v_5, v_6)$, $C = (v_1, v_2, v_5, v_6, v_7)$, $C = (v_1, v_2, v_5, v_6, v_7, v_4)$, $C = (v_1, v_2, v_5, v_6, v_7, v_4, v_3)$, и наконец, последовательность $C = (v_1, v_2, v_5, v_6, v_7, v_4, v_3, v_1)$, которая представляет искомый гамильтонов цикл.

На рисунке 4.25 выделены ребра, принадлежащие полученному гамильтонову циклу. Таким же способом, начиная от каждой вершины графа, можно попытаться построить гамильтонову цепь.

Обобщением задачи поиска гамильтонова цикла является известная *задача коммивояжера*. Пусть имеется несколько городов, расстояния между которыми заданы. Коммивояжер должен посетить все города по одному разу и вернуться в тот, с которого начал обход. При этом требуется выбрать такой маршрут движения коммивояжера, при котором суммарное пройденное расстояние будет минимальным. Графовой моделью этой задачи является следующая: задан полный граф с взвешенными ребрами (весами служат расстояния), требуется найти такой гамильтонов цикл, что сумма весов всех его ребер была бы минимальной.

4.3.4 Кратчайшие пути в графе

Задан связный взвешенный орграф $G = (V, A)$, каждой дуге $a = (v_i, v_j)$ которого приписано положительное число, называемое длиной дуги $l(a) = l(v_i, v_j)$ (рисунок 4.26). Длина пути во взвешенном графе равна сумме длин дуг, входящих в этот путь.

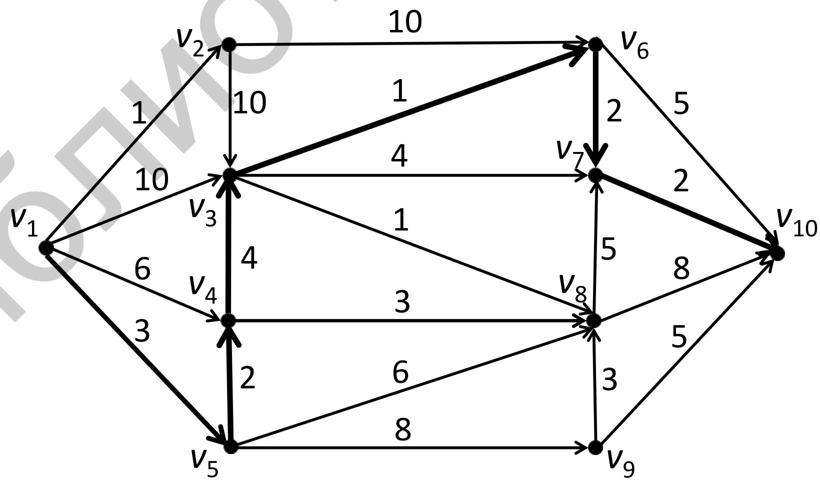


Рисунок 4.26 – Взвешенный орграф с выделенным кратчайшим путем

Задача о кратчайшем пути состоит в отыскании в орграфе пути G, связывающего две заданные вершины (начальную v_0 и конечную v_k) и имеющего минимальную длину при условии, что хотя бы один такой путь существует.

Для решения этой задачи можно применить *алгоритм Дейкстры*, предложенный нидерландским ученым Э. Дейкстрой в 1959 г. Алгоритм основан на распространении волны путей от начальной вершины. В результате просмотра всех путей от начальной вершины v_0 находится длина кратчайшего пути до конечной вершины v_k. В результате обратного просмотра находится сам кратчайший путь.

Согласно алгоритму Дейкстры каждой вершине из $v \in \boldsymbol{V}$ сопоставляется метка $\lambda(v)$, равная расстоянию от вершины v_0 до v. В процессе работы алгоритма метка может быть постоянной или временной. В первом случае она равна длине минимального (v_0, v)-пути, во втором – длине найденного пути от v_0, проходящего только через вершины с постоянными метками. Перед началом работы алгоритма начальной вершине приписывается метка $\lambda(v_0) = 0$, остальным вершинам – временные метки, равные $\lambda(v_i) = \infty$.

На каждом шаге алгоритма выбирается для рассмотрения одна вершина v с наименьшей временной меткой $\lambda(v)$ и делается попытка уменьшить временные метки вершин из ее полуокрестности исхода Γ^+v (вершины с постоянными метками не рассматриваются). Если для очередной рассматриваемой вершины $u \in \Gamma^+v$ выполняется $\lambda(v) > \lambda(v) + l(v, u)$, то метка $\lambda(u)$ этой вершины заменяется на $\lambda(v) + l(v, u)$ (иначе она не изменяется) и считается, что она получила метку из вершины v. После просмотра вершин из Γ^+v метка вершины v объявляется постоянной. Работа алгоритма завершается, когда метка конечной вершины v_k станет постоянной. В результате выполнения описанной процедуры длина кратчайшего (v_0, v_k)-пути в графе $G = (V, A)$ определяется равной метке $\lambda(v_k)$.

Пройденный путь можно найти, двигаясь от конечной вершины v_k к начальной v_0. При этом всякий раз после вершины v надо выбирать для включения в искомый путь такую вершину u, чтобы выполнялось равенство $\lambda(v) - \lambda(u) = l(u, v)$.

Для иллюстрации алгоритма найдем в заданном графе (см. рисунок 4.26) кратчайший (v_1, v_{10})-путь. Покажем изменение меток вершин в процессе изменения меток вершин от v_1 к вершине v_{10}. Ниже слева приведена выбираемая на соответствующем шаге вершина с наименьшей временной меткой (объявляемая на этом шаге постоянной), за ней – текущие значения временных меток вершин. Постоянные метки вершин не показываются. При реализации алгоритма реализуется девять шагов, соответствующие последовательности изменения временных меток вершин имеют следующий вид:

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
	0	∞	∞	∞	∞	∞	∞	∞	∞	∞
v_1:		1	10	6	3	∞	∞	∞	∞	∞
v_2:			10	6	3	11	∞	∞	∞	∞
v_5:			10	5		11	∞	9	11	∞
v_4:			9			11	∞	8	11	∞
v_8:			9			11	13		11	16
v_3:						10	13		11	16
v_6:							12		11	15
v_9:							12			15
v_7:										14

Длина кратчайшей цепи в данном графе равна 14. Двигаясь от вершины v_{10} к вершине v_1, сначала включаем в искомый путь вершину v_7, так как для нее $\lambda(v_{10}) - \lambda(v_7) = l(v_7, v_{10}) = 2$. Затем, следуя тому же правилу, проходим вершины v_6, v_3, v_4, v_5 и v_1. Таким образом находится путь $v_1, v_5, v_4, v_3, v_6, v_7, v_{10}$, связывающий вершины v_1 и v_{10} и имеющий наименьшую длину (14 в данном случае). На рисунке 4.26 выделены ребра графа, принадлежащие найденному пути.

Алгоритм Дейкстры можно применить к неориентированным (и смешанным) взвешенным графам. Для этого достаточно заметить каждое ребро (v, u) парой дуг (v, u) и (u, v), имеющих ту же длину.

4.4 Независимость и покрытия

Классическими задачами теории графов являются задачи нахождения наибольших множеств элементов (вершин или ребер), смежных (или инцидентных) со всеми остальными или попарно несмежных. Вводятся понятия независимости, доминирования на множестве однородных элементов (вершин или ребер) и покрытия – на множестве разнородных элементов. С этими понятиями связаны числа вершинного и реберного покрытий, независимости и паросочетания, которые являются инвариантами графа.

4.4.1 Доминирующие множества графа

Подмножество D множества вершин V графа $G = (V, E)$ называется *доминирующим* (внешне устойчивым) множеством графа G, если выполняется условие $D \cup \Gamma D = V$, где $\Gamma D = \bigcup_{v \in D} \Gamma v$. Другими словами, множество D является *доминирующим множеством* графа G, если каждая вершина из множества $V \setminus D$ смежна с некоторой вершиной из D.

Очевидно, что если D является доминирующим множеством некоторого графа G, то всякое множество вершин $D' \supseteq D$ этого графа также является доминирующим, поэтому представляет интерес задача нахождения *минимальных* доминирующих множеств, т. е. таких множеств, ни одно собственное подмножество которых не является доминирующим. Доминирующее множество, имеющее наименьшую мощность, принято называть *наименьшим*. Мощность наименьшего доминирующего множества называется *числом доминирования* графа G и является его инвариантом.

Например, для графа на рисунке 4.27 множества $D_1 = \{v_2, v_3, v_5, v_6\}$, $D_2 = \{v_1, v_4, v_7\}$, $D_3 = \{v_3, v_5, v_8, v_9\}$, $D_4 = \{v_3, v_5, v_7, v_9\}$, $D_5 = \{v_2, v_4, v_7\}$, $D_6 = \{v_2, v_6\}$ являются доминирующими, причем все множества, кроме D_1, минимальные, а D_6 – наименьшее доминирующее множество.

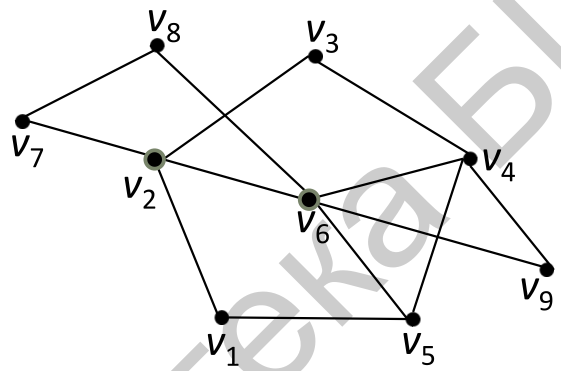


Рисунок 4.27 – Граф с выделенным наименьшим доминирующим множеством

Примером задачи о наименьшем доминирующем множестве является одна из задач о ферзях, где надо расставить на шахматной доске наименьшее число ферзей так, чтобы каждая клетка была под ударом хотя бы одного из них. Для этого достаточно найти наименьшее доминирующее множество в графе, вершины которого соответствуют клеткам шахматной доски и две вершины связаны ребром, если и только если они взаимно достижимы ходом ферзя. Найденное наименьшее доминирующее множество вершин указывает, куда надо поставить ферзей, а их число, которое в данном случае равно 5, есть число доминирования данного графа (рисунок 4.28).

Задача о наименьшем доминирующем множестве может быть сведена к задаче о кратчайшем покрытии булевой матрицы смежности заданного графа G. Для этого матрицу смежности дополним единицами на главной диагонали (дополнив отношение смежности на множестве вершин графа до рефлексивного). Находим такую наименьшую совокупность строк полученной матрицы, чтобы каждый ее столбец имел единицу, по крайней мере, в одной из строк найденной совокупности. Множество вершин графа, которым соответствуют строки найденной совокупности, представляет наименьшее доминирующее множество.

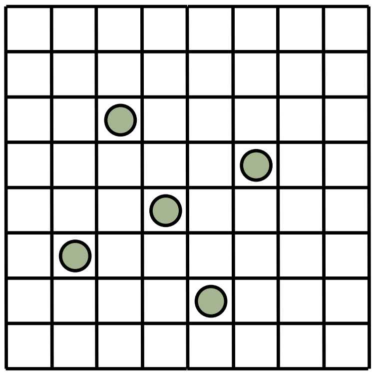


Рисунок 4.28 – Решение задачи о пяти ферзях

Найдем наименьшее доминирующее множество графа $G = (V, E)$, показанного на рисунке 4.27. Матрица смежности этого графа, дополненная единицами на главной диагонали, имеет вид

v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	
1	1	0	0	1	0	0	0	0	Γv_1
1	1	1	0	0	1	1	0	0	Γv_2
0	1	1	1	0	0	0	0	0	Γv_3
0	0	1	1	1	1	0	0	1	Γv_4
1	0	0	1	1	1	0	0	0	Γv_5
0	1	0	1	1	1	0	1	1	Γv_6
0	1	0	0	0	0	1	1	0	Γv_7
0	0	0	0	0	1	1	1	0	Γv_8
0	0	0	1	0	1	0	0	1	Γv_9

Столбцы v_4 и v_6 могут быть удалены из матрицы, так как они поглощают столбец v_9, после их удаления появляется возможность удалить строки Γv_5, Γv_8 и Γv_9, так они поглощаются строками Γv_1, Γv_7 и Γv_6 соответственно. После этих удалений матрица приобретает следующий вид:

v_1	v_2	v_3	v_5	v_7	v_8	v_9	
1	1	0	1	0	0	0	Γv_1
1	1	1	0	1	0	0	Γv_2
0	1	1	0	0	0	0	Γv_3
0	0	1	1	0	0	1	Γv_4
0	1	0	1	0	1	1	Γv_6
0	1	0	0	1	1	0	Γv_7

Столбцы v_2 и v_5 удаляются из матрицы, так как они поглощают столбцы v_1 и v_9 соответственно. После удаления столбцов появляется возможность исключить из матрицы строки Γv_1 и Γv_3, так как они поглощаются строкой Γv_2. После этих удалений матрица приобретает следующий вид:

v_1	v_3	v_7	v_8	v_9	
1	1	1	0	0	Γv_2
0	1	0	0	1	Γv_4
0	0	0	1	1	Γv_6
0	0	1	1	0	Γv_7

Нетрудно заметить, что наименьшее покрытие этой матрицы состоит из строк Γv_2 и Γv_6, порождающих наименьшее доминирующее множество $D = \{v_2, v_6\}$ графа G.

4.4.2 Независимые множества графа

Множество $S \subseteq V$ вершин графа $G = (V, E)$ называется *независимым* (*внутренне устойчивым*) множеством графа G, если выполняется условие $S \cap \Gamma S = \varnothing$, где $\Gamma S = \bigcup\limits_{v \in S} \Gamma v$, а Γv – окрестность вершины v.

Другими словами, множество вершин графа является независимым (рисунок 4.29), если никакие две из них не смежны. Очевидно, что если S – независимое множество, то любое его подмножество также является независимым. Поэтому представляет интерес задача нахождения в графе *максимальных* (по включению) *независимых множеств*, т. е. таких, которые не являются собственными подмножествами никаких других независимых множеств. Среди максимальных независимых множеств выделяют множество графа G, имеющее наибольшую мощность, его называют *наибольшим независимым множеством*. Мощность наибольшего независимого множества называется *вершинным числом независимости* (числом внутренней устойчивости, неплотностью) графа G и обозначается символом $\beta_0(G)$.

Необходимо отметить, что задачи, связанные с отысканием наибольшего независимого множества и числа независимости, в общем случае трудны для алгоритмического решения. Однако для некоторых графов их нахождение тривиально. Например, число независимости полного графа $\beta_0(K_p) = 1$, полного двудольного графа $\beta_0(K_{p,q}) = \max(p, q)$.

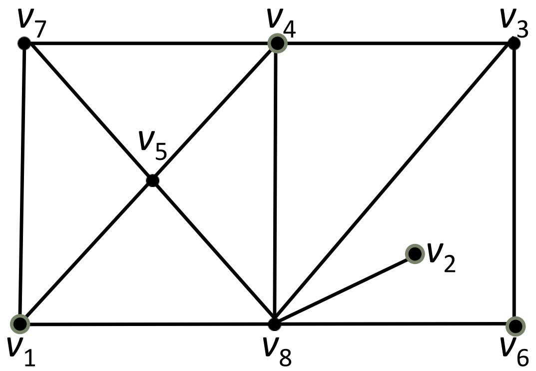


Рисунок 4.29 – Граф с выделенным наибольшим независимым множеством

Примером задачи нахождения наибольшего независимого множества является задача о восьми ферзях (Гаусса): можно ли расставить на шахматной доске восемь ферзей так, чтобы ни один из них не находился под ударом другого. Эта задача сводится к нахождению наибольшего независимого множества графа с 64 вершинами (рисунок 4.30), в котором две вершины смежны, если ферзи, стоящие на соответствующих клетках шахматной доски, находятся под ударом друг друга.

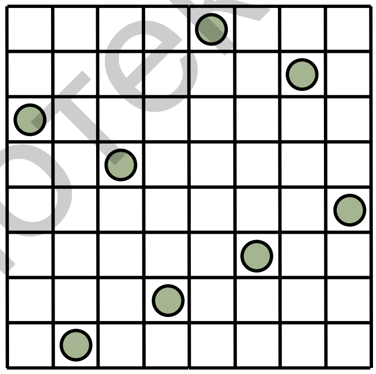


Рисунок 4.30 – Решение задачи о восьми ферзях

Проверка независимого множества на максимальность тривиальна, благодаря следующему его свойству: независимое множество S является максимальным в том и только в том случае, когда $S \cup \Gamma S = V$. Действительно, если это равенство не выполняется, то в множестве $V \setminus S$ найдется вершина, не смежная ни с одной вершиной из S. Присоединив ее к S, получим независимое множество, содержащее S в качестве собственного подмножества, что противоречит предположению о максимальности S. Если же это равенство выполняется, то не существует вершины в графе G, которая была бы не смежна ни с одной вершиной из S и ее можно было бы добавить к S, не нарушая независимости S.

Второй способ проверки независимого множества вершин на максимальность следует из тесной связи понятий независимого и доминирующего множеств: независимое множество является максимальным тогда только тогда, когда оно является доминирующим. Это утверждение следует из определений этих понятий и может быть доказано от противного. Если максимальное независимое множество S не является доминирующим, тогда из определения доминирования следует, что существует вершина $v_i \notin S$, не смежная ни с одной его вершиной. Но тогда ее можно включить в S, что противоречит предположению о максимальности S. И обратно: если предположить, что независимое доминирующее множество S не является максимальным, то из этого следует, что найдется некоторая вершина $v_i \notin S$, не смежная ни с одной вершиной из S, что противоречит тому, что множество S является доминирующим. Например, легко убедиться, что наибольшее независимое множество $S = \{v_2, v_6, v_1, v_4\}$ графа на рисунке 4.29 является доминирующим.

Один из способов нахождения в графе всех максимальных независимых множеств S_i заключается в последовательном наращивании числа вершин в этих множествах. Если очередная вершина не смежна не со всеми вершинами некоторого множества S_k, то на основе S_k формируется новое множество S_{k+1}, не включающее вершины из S_k, смежные с вновь добавляемой. Из всех множеств S_i на каждом шаге сохраняются только максимальные (не являющиеся собственными подмножествами других).

Для графа, приведенного на рисунке 4.29, в качестве начального значения первого формируемого максимального независимого множества принимаем $S_1 = \{v_1\}$. Вершину v_2 можно также добавить в множество S_1, так как она не смежна с v_1. К ним можно добавить также и вершину v_3, получая $S_1 = \{v_1, v_2, v_3\}$. Следующая вершина v_4 не смежна с v_1 и v_2, но смежна с v_3, поэтому, наряду с $S_1 = \{v_1, v_2, v_3\}$, образуем новое множество $S_2 = \{v_1, v_2, v_4\}$.

Приведем последовательность дальнейших преобразований (с указанием выбираемых вершин) формируемых максимальных независимых множеств:

$\{\{v_1, v_2, v_3\}, \{v_1, v_2, v_4\}\}$;

$v_5 : \{\{v_1, v_2, v_3\}, \{v_2, v_3, v_5\}, \{v_1, v_2, v_4\}\}$;

$v_6 : \{\{v_1, v_2, v_3\}, \{v_2, v_3, v_5\}, \{v_2, v_5, v_6\}, \{v_1, v_2, v_4, v_6\}\}$;

$v_7 : \{\{v_1, v_2, v_3\}, \{v_2, v_3, v_7\}, \{v_2, v_6, v_7\}, \{v_2, v_3, v_5\}, \{v_2, v_5, v_6\}, \{v_1, v_2, v_4, v_6\}\}$;

$v_8 : \{\{v_1, v_2, v_3\}, \{v_2, v_3, v_7\}, \{v_7,v_8\}, \{v_2,v_6, v_7\}, \{v_2, v_3, v_5\}, \{v_2, v_5, v_6\}, \{v_1,v_2,v_4,v_6\}\}$.

Таким образом, получено семь максимальных независимых множеств, наибольшим является множество $\{v_1, v_2, v_4, v_6\}$. Вершинное число независимости анализируемого графа $\beta_0(G) = 4$.

Во многих практических приложениях теории графов требуется найти только одно наибольшее независимое множество. Для этого можно было бы получить все максимальные независимые множества и из них выбрать наибольшее. Однако в теории графов доказано, что число максимальных независимых множеств растет во много раз быстрее, чем число вершин графа. Поэтому представляет интерес метод нахождения наибольшего независимого множества, не требующий получения всех максимальных.

Поиск наибольшего независимого множества графа, имеющего n вершин, будем осуществлять в процессе построения максимальных независимых множеств и разбивать на этапы, каждый из которых связан с определенной вершиной $v_i \in V$ графа. На i-м этапе находятся максимальные независимые множества, содержащие вершину v_i и не содержащие вершин с меньшими номерами, т. е. таких v_j, для которых $j < i$.

Пусть S_i – множество вершин, образующих независимое множество, формируемое на i-м этапе, его начальное значение состоит из единственной вершины v_i. Множество S_i последовательно расширяется за счет поочередного включения в него $v_j \in V$ $(j > i)$, не смежных ни с одной вершиной из S_i, т. е. вершин, удовлетворяющих следующим условиям:

$$i < j \le n \quad \text{и} \quad v_j \in \bigcap_{v \in S_i} \{V \setminus \Gamma v\}. \tag{4.1}$$

Такое расширение множества S_i продолжается до тех пор, пока множество $\bigcap_{v \in S_i} \{V \setminus \Gamma v\}$ не станет пустым.

Независимое множество, соответствующее результату расширения S_i, проверяется на максимальность в соответствии со следующим свойством: независимое множество S_i является максимальным только в том случае, когда

$$S_i \cup \Gamma S_i = V.$$

Действительно, если это равенство не выполняется, то в множестве $V \setminus S_i$ найдется вершина, не смежная ни с одной вершиной $v \in S_i$. Эту вершину можно добавить в S_i, в результате чего получим независимое множество, которое содержит S_i в качестве собственного подмножества, что означает, что S_i не максимально. Если же объединение множеств $S_i \cup \Gamma S_i$ совпадает со всем множеством V, то в графе не существует вершины, которая была бы не смежна ни с одной вершиной из S_i и ее можно было бы добавить к S_i, не нарушив его независимость.

Множество, прошедшее такую проверку, включается в решение. Чтобы на i-м этапе построить следующее независимое множество, из полученного к этому шагу множества S_i (включаемого или не включаемого в решение) удаляется вершина v_p, присоединенная к S_i последней, и выполняется та же процедура с вершинами v_q, где $q > p$.

За n этапов можно получить все максимальные независимые множества графа и выбрать из них наибольшее. Однако если необходимо построить только одно наибольшее независимое множество, то можно сократить перебор максимальных множеств независимости за счет исключения из рассмотрения тех, которые заведомо не приведут к большему максимальному множеству независимости, чем уже полученное. Пусть наибольшее из найденных независимых множеств имеет мощность l. Будем использовать следующие очевидные правила, ведущие к сокращению поиска:

1) для каждой вершины $v_i \in V$ найти множества не смежных с ней вершин $I(v_i) = V \setminus \Gamma v_i$;

2) упорядочить вершины по убыванию значений $|I(v_i)|$ (или, что то же, по возрастанию степеней), получив список V^*, для того чтобы как можно раньше найти искомое независимое множество;

3) найти верхнюю оценку мощности независимого множества, которое может быть получено на текущем этапе;

4) после выполнения каждого этапа исключать из списка V^* (и множеств $I(v_i)$ несмежности его вершин):

– вершины, уже рассмотренные в качестве начальных при формировании независимых множеств;

– вершины v_i, для которых $|I(v_i)| \leq l$;

5) закончить процесс формирования независимых множеств тогда, когда:

– текущее независимое множество имеет мощность, равную верхней оценке;

– невозможно построить большее независимое множество, чем уже найденное, т. е. когда выполняется условие: $|V^*| \leq l$.

При поиске наибольшего независимого множества графа G иногда полезно иметь верхнюю оценку его числа независимости $\beta_0(G)$, которую можно получить из следующих соображений: для того чтобы в графе существовало независимое множество мощности k необходимо (но недостаточно), чтобы в графе имелось не менее k вершин, для которых $|I(v_i)| \geq (k–1)$.

Пусть задан рассмотренный выше граф $G = (V, E)$ (см. рисунок 4.29). Для вершин $v_i \in V$ находим множества $I(v_i) = V \setminus \Gamma v_i$ не смежных им вершин:

$I(v_1) = \{v_2, v_3, v_4, v_6\}$, $|I(v_1)| = 4$;

$I(v_2) = \{v_1, v_3, v_4, v_5, v_6, v_7\}$, $|I(v_2)| = 6$;

$I(v_3) = \{v_1, v_2, v_5, v_7\}$, $|I(v_3)| = 4$;

$I(v_4) = \{v_1, v_2, v_6\}$, $|I(v_4)| = 3$;

$I(v_5) = \{v_2, v_3, v_6\}$, $|I(v_5)| = 3$;

$I(v_6) = \{v_1, v_2, v_4, v_5, v_7\}$, $|I(v_6)| = 5$;

$I(v_7) = \{v_2, v_3, v_6, v_8\}$, $|I(v_7)| = 4$;

$I(v_8) = \{v_7\}$, $|I(v_8)| = 1$.

Будем выбирать для формирования независимых множеств в качестве начальных вершины из множества $V^* = \{v_2, v_6, v_1, v_3, v_7, v_4, v_5, v_8\}$, упорядоченного по убыванию мощностей 6, 5, 4, 4, 4, 3, 3, 1 их множеств несмежности. Очевидно, что верхняя оценка мощности наибольшего независимого множества равна 5, так как в графе имеется только 5 вершин, для которых $|I(v_i)| \geq 4$.

На первом этапе алгоритма выбираем из V^* первую вершину – v_2 и в качестве S_1 принимаем множество $\{v_2\}$, которое не является пока максимальным независимым множеством.

Далее последовательно выбираем следующие вершины из V^*:

– v_6, получая $S_1 = \{v_2, v_6\}$ и $l = 2$;

– v_1, получая $S_1 = \{v_2, v_6, v_1\}$ и $l = 3$;

– v_4 (v_3 и v_7 не удовлетворяет условию (4.1)), получая $S_1 = \{v_2, v_6, v_1, v_4\}$ и $l = 4$.

На этом процесс поиска максимального независимого множества, включающего вершину v_2, обрывается, так как попытки исключения и замены некоторых вершин из текущего множества S_1 на другие вершины из V^* терпят неудачу.

После выполнения первого этапа множества $I(v_i)$ вершин $v_i \in V^*$ корректируются в соответствии с вышеупомянутыми правилами, в результате чего получаем

$I(v_6) = \{v_1, v_4, v_5, v_7\}$, $I(v_1) = \{v_3, v_4, v_6\}$, $I(v_3) = \{v_1, v_5, v_7\}$, $I(v_7) = \{v_3, v_6, v_8\}$,

$I(v_4) = \{v_1, v_6\}$, $I(v_5) = \{v_3, v_6\}$, $I(v_8) = \{v_7\}$,

$|I(v_6)| = 4$, $|I(v_1)| = 3$, $|I(v_3)| = 3$, $|I(v_7)| = 3$, $|I(v_4)| = 2$, $|I(v_5)| = 2$, $|I(v_8)| = 1$.

Соответственно множество V^* сокращается до $V^* = \{v_6\}$, так как только $|I(v_6)| \geq 4$, что завершает процесс поиска наибольшего независимого множества. Искомым наибольшим независимым множеством является $S_1 = \{v_2, v_6, v_1, v_4\}$.

4.4.3 Клики графа

Понятие клики графа является антиподом понятия независимого множества графа. *Кликой* графа называется множество вершин, которое порождает полный подграф. Другими словами, любые две вершины, входящие в клику, смежны. Клика, не являющаяся подмножеством никакой другой клики (с большим числом вершин), называется *максимальной*, а клика с наибольшим числом вершин – *наибольшей*. Число вершин в наибольшей клике графа G называется его *кликовым числом* (или *плотностью*) $\varphi(G)$.

Задача о клике относится к классу труднорешаемых задач теории графов, алгоритма, решающего эту задачу за полиномиальное время, пока не найдено. Известны две основные формулировки задачи о клике: задача распознавания, когда требуется определить, существует ли в заданном графе клика заданного размера, и вычислительная задача нахождения наибольшей клики в заданном графе (и соответственно кликового числа графа). К задаче о нахождении множества максимальных клик или наибольшей клики графа сводятся многие задачи теории графов, логического проектирования, биоинформатики, химии и др. Разработан ряд алгоритмов, обеспечивающих получение всех максимальных клик или наибольшей клики, а также приближенные алгоритмы, результатом работы которых является максимальная, но не обязательно наибольшая клика.

Известный алгоритм Брона – Кербоша построения максимальных клик неориентированного графа основан на том факте, что клика является максимальным по включению полным подграфом. В основе алгоритма лежит процедура наращивания числа вершин в кликах (аналогично тому, как это делается при построении максимальных независимых множеств): начиная с одной вершины, на каждом шаге алгоритма делается попытка увеличить уже построенный полный подграф, добавляя в него одну из вершин-кандидатов на включение.

Ниже приводится алгоритм А. Д. Закревского для нахождения в графе совокупности S всех максимальных клик. Алгоритм основан на последовательном сокращении числа вершин в подмножествах-кандидатах $S_l \in S$ на образование максимальной клики. В качестве начального значения совокупности S принимается $S = \{V\}$. Далее на i-м шаге выбирается некоторая очередная нерассмотренная вершина v, строится ее окрестность Γv и для каждого из подмножеств S_j из текущего состояния совокупности $S = \{S_1, S_2, \ldots, S_l\}$ подмножеств-кандидатов проверяются условия

$$v \in S_j \text{ и } S_j \cap (V \setminus (\Gamma v \cup \{v\})) \neq \varnothing . \tag{4.2}$$

При выполнении этих условий находятся множества

$$S_j^1 = S_j \setminus \{v\};\ \ S_j^2 = S_j \cap (\Gamma v \cup \{v\}),$$

подмножество S_j исключается из текущей совокупности S, множества S_j^1 и S_j^2 включаются в S, если они не являются подмножествами ни одного из других подмножеств $S_k \in S$.

После перебора всех вершин графа (кроме последней, для которой приведенные условия уже не будут выполняться) совокупность S будет представлять множество всех максимальных клик графа.

Работа алгоритма иллюстрируется на примере нахождения максимальных клик графа, приведенного на рисунке 4.31.

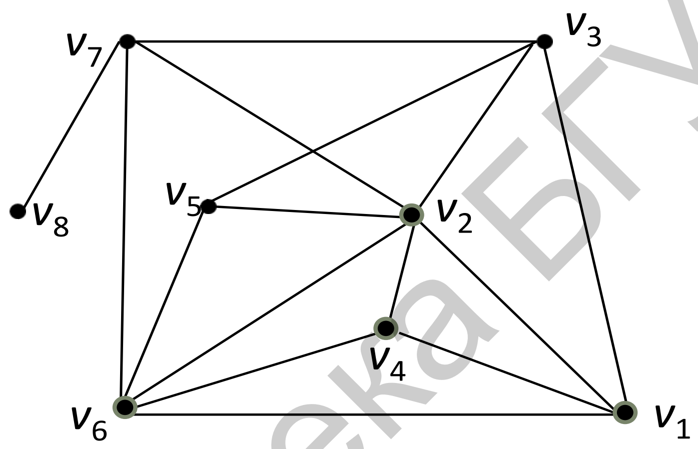


Рисунок 4.31 – Граф с выделенной наибольшей кликой

Для удобства проверки условий (4.2) выпишем предварительно окрестности вершин графа:

$\Gamma v_1 = \{v_2, v_3, v_4, v_6\}$;

$\Gamma v_2 = \{v_1, v_3, v_4, v_5, v_6, v_7\}$;

$\Gamma v_3 = \{v_1, v_2, v_5, v_7\}$;

$\Gamma v_4 = \{v_1, v_2, v_6\}$;

$\Gamma v_5 = \{v_2, v_3, v_6\}$;

$\Gamma v_6 = \{v_1, v_2, v_4, v_5, v_7\}$;

$\Gamma v_7 = \{v_2, v_3, v_6, v_8\}$;

$\Gamma v_8 = \{v_7\}$.

Ниже приведена последовательность состояний совокупности S, представляющих результаты ее преобразований при выборе вершин v_1, v_2, v_3, v_4, v_5,

v_6, v_7, начиная с начального состояния $S = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$. Полужирным шрифтом выделены те подмножества из S, для которых выполняется условие (4.2) для следующей выбираемой переменной. Зачеркнутые подмножества поглощаются другими и подлежат удалению.

$S = \{\boldsymbol{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8}\}$;

v_1: $\{\{v_1, v_2, v_3, v_4, v_6\}, \{\boldsymbol{v_2, v_3, v_4, v_5, v_6, v_7, v_8}\}\}$;

v_2: $\{\{\boldsymbol{v_1, v_2, v_3, v_4, v_6}\}, \{\boldsymbol{v_3, v_4, v_5, v_6, v_7, v_8}\}, \{\boldsymbol{v_2, v_3, v_4, v_5, v_6, v_7}\}\}$;

v_3: $\{\{v_1, v_2, v_4, v_6\}, \{v_1, v_2, v_3\}, \{\boldsymbol{v_4, v_5, v_6, v_7, v_8}\}, \{\cancel{v_3, v_5, v_7}\}, \{\boldsymbol{v_2, v_4, v_5, v_6, v_7}\}, \{v_2, v_3, v_5, v_7\}\}$;

v_4: $\{\{v_1, v_2, v_4, v_6\}, \{v_1, v_2, v_3\}, \{\boldsymbol{v_5, v_6, v_7, v_8}\}, \{\cancel{v_4, v_6}\}, \{\boldsymbol{v_2, v_5, v_6, v_7}\}, \{\cancel{v_2, v_4, v_6}\}, \{\boldsymbol{v_2, v_3, v_5, v_7}\}\}$;

v_5: $\{\{v_1, v_2, v_4, v_6\}, \{v_1, v_2, v_3\}, \{\boldsymbol{v_6, v_7, v_8}\}, \{\cancel{v_5, v_6}\}, \{v_2, v_6, v_7\}, \{v_2, v_5, v_6\}, \{v_2, v_3, v_7\}, \{v_2, v_3, v_5\}\}$;

v_6: $\{\{v_1, v_2, v_4, v_6\}, \{v_1, v_2, v_3\}, \{v_7, v_8\}, \{\cancel{v_6, v_7}\}, \{v_2, v_6, v_7\}, \{v_2, v_5, v_6\}, \{v_2, v_3, v_7\}, \{v_2, v_3, v_5\}\}$;

v_7: $\{\{v_1, v_2, v_4, v_6\}, \{v_1, v_2, v_3\}, \{v_7, v_8\}, \{v_2, v_6, v_7\}, \{v_2, v_5, v_6\}, \{v_2, v_3, v_7\}, \{v_2, v_3, v_5\}\}$.

Последняя строка представляет множество всех максимальных клик анализируемого графа (см. рисунок 4.31). Наибольшей кликой графа является $\{v_1, v_2, v_4, v_6\}$, соответственно кликовое число этого графа равно 4.

Во многих практических приложениях теории графов требуется найти только одну наибольшую клику. Казалось бы, что для этого можно, воспользовавшись предложенным методом, получить все максимальные клики и из них выбрать наибольшую. Но в теории графов показано, что число максимальных клик растет во много раз быстрее, чем число вершин графа. Поэтому представляет интерес метод нахождения наибольшей клики, не требующий получения всех максимальных клик. При поиске наибольшей клики графа иногда полезно иметь верхнюю оценку его кликового числа, которая получается из следующих очевидных соображений: для того чтобы в графе существовала клика мощности k, необходимо (но недостаточно), чтобы в графе имелось не менее k вершин со степенями не менее $(k - 1)$.

Рассмотрим метод нахождения максимальных клик графа, аналогичный методу поиска наибольшего независимого множества, приведенному в пункте 4.1.4. Процесс нахождения максимальных клик графа, имеющего n вершин, разбивается на этапы, каждый из которых связан с определенной вершиной $v_i \in V$ графа. На i-м этапе находятся все максимальные клики, содержащие вершину v_i и не содержащие вершин с меньшими номерами, т. е. таких v_j, для которых $j < i$.

Пусть S_i – множество вершин, образующих одну из клик, формируемых на i-м этапе. За начальное значение множества S_i принимается множество, состоящее из единственной вершины v_i. Далее множество S_i последовательно расширяется за счет поочередного включения в него элементов $v_j \in V$, удовлетворяющих следующим условиям:

$$i < j \leq n \quad \text{и} \quad v_j \in \bigcap_{v \in S_i} \Gamma v. \tag{4.3}$$

Каждый раз при соблюдении этих условий выбирается вершина v_j с минимальным номером j. Такое расширение множества S_i продолжается до тех пор, пока множество $\bigcap_{v \in S_i} \Gamma v$ не станет пустым.

Клика, соответствующая результату расширения S_i, проверяется на максимальность согласно следующему свойству: клика S_i является максимальной в том и только в том случае, когда

$$\bigcup_{v \in S_i} (V \setminus \Gamma v) = V.$$

Действительно, если это равенство не выполняется, то в множестве $V \setminus S_i$ найдется вершина, входящая в окрестности Γv всех вершин $v \in S_i$. Эту вершину можно добавить в S_i, в результате чего получим клику, которая содержит S_i в качестве собственного подмножества, что означает, что S_i не максимально. Если же это объединение множеств $V \setminus \Gamma v$ совпадает со всем множеством V, то для любой вершины, не входящей в S_i, в S_i найдется хотя бы одна вершина, не смежная с ней. Следовательно, клика S_i является максимальной, так как в нее нельзя добавить ни одной вершины, не нарушив условие попарной смежности ее вершин.

Клика, прошедшая такую проверку, включается в решение. Чтобы на i-м этапе построить следующую клику, из полученного множества S_i (включенного или не включенного в решение) удаляется вершина v_p, присоединенная к S_i последней, и выполняется та же процедура с вершинами v_q, где $q > p$.

За n этапов можно получить все максимальные клики графа и выбрать наибольшую. Однако если необходимо построить только одну наибольшую клику, то можно сократить перебор максимальных клик за счет исключения из рассмотрения тех, которые заведомо не приведут к большей максимальной клике, чем уже полученная. Пусть мощность текущей наибольшей клики равна l.

Будем использовать следующие очевидные правила, ведущие к сокращению перебора клик графа:

1) упорядочить вершины по убыванию их степеней, получив список V^*, для того, чтобы как можно раньше найти искомую клику;

2) найти верхнюю оценку мощности наибольшей клики;

3) после выполнения каждого этапа исключать из списка V^* (и окрестностей его вершин):

– вершины, уже рассмотренные в качестве начальных при формировании клик;

– вершины, степени которых не превышают l;

4) закончить процесс формирования клик тогда, когда:

– текущая клика имеет мощность, равную ее верхней оценке, и тогда наибольшая клика найдена;

– невозможно построить бóльшую клику, чем уже найденная, т. е. когда выполняется условие $|V^*| \leq l$.

Пусть задан рассмотренный ранее граф (см. рисунок 4.31). Будем выбирать для формирования клик в качестве начальных вершины из множества $V^* = \{v_2, v_6, v_1, v_3, v_7, v_4, v_5, v_8\}$, упорядоченного по убыванию степеней 6, 5, 4, 4, 4, 3, 3, 1 вершин графа. Очевидно, что верхняя оценка мощности наибольшей клики равна 5.

На первом этапе алгоритма выбираем из V^* первую вершину – v_2 и в качестве S_1 принимаем множество $\{v_2\}$, которое не является пока максимальной кликой. Далее последовательно выбираем следующие вершины из V^*:

– v_6, получая $S_1 = \{v_2, v_6\}$ и $l = 2$;

– v_1, получая $S_1 = \{v_2, v_6, v_1\}$ и $l = 3$;

– v_4 (v_3 и v_7 не удовлетворяет условию (4.3)), получая $S_1 = \{v_2, v_6, v_1, v_4\}$ и $l = 4$.

На этом процесс поиска максимальной клики, включающей вершину v_2, обрывается, так как попытки исключения и замены некоторых вершин из текущего множества S_1 на другие вершины из V^* терпят неудачу.

После выполнения первого этапа степени вершин множества V^* корректируются в соответствии с вышеупомянутыми правилами:

$$d(v_6) = 4,\ d(v_1) = 3,\ d(v_3) = 3,\ d(v_7) = 3,\ d(v_4) = 2,\ d(v_5) = 2,\ d(v_8) = 1.$$

Соответственно множество V^* сокращается до $V^* = \{v_6\}$, что обусловливает завершение процесса поиска наибольшей клики. Искомой наибольшей кликой является множество вершин $S_1 = \{v_2, v_6, v_1, v_4\}$.

4.4.4 Независимые множества и клики графа

Задача нахождения клик графа и задача нахождения независимых множеств в некотором графе являются двойственными по отношению друг к другу.

Очевидно, что задача о независимом множестве преобразуется в задачу о клике и наоборот простым переходом. Подграф графа G, порождаемый его независимым множеством, является пустым подграфом (множество его ребер пусто). В графе $\overline{G}$, который является дополнением графа G, независимое множество вершин графа G порождает полный подграф (который также иногда называют кликой). То же самое можно сказать и о клике графа G: вершины, входящие в нее, порождают в графе $\overline{G}$ пустой подграф, определяющий независимое множество вершин графа $\overline{G}$.

Отсюда следует, что:

– задачи нахождения клик и независимых множеств графа сводятся друг к другу;

– число независимости графа G равно кликовому числу дополнительного графа $\overline{G}$: $\beta_0(G) = \varphi(\overline{G})$.

К примеру, в пункте 4.4.3 было найдено, что наибольшей кликой графа G, приведенного на рисунке 4.31, является множество $\{v_1, v_2, v_4, v_6\}$ вершин, а в пункте 4.4.2 показано, что это же множество является наибольшим независимым множеством графа, который показан на рисунке 4.29 и который является дополнением графа G.

4.4.5 Вершинные покрытия графа

Далее будем считать, что вершина *покрывает* инцидентные ей ребра, а ребро покрывает инцидентные ему вершины. Множество вершин графа, покрывающих все его ребра, называется *вершинным покрытием* графа. Другими словами, множество $C \subseteq V$ вершин графа $G = (V, E)$ называется его вершинным покрытием (рисунок 4.32), если каждое ребро из E покрывается хотя бы одной вершиной из C.

Вершинное покрытие графа называется *минимальным*, если оно не содержит покрытия с меньшим числом вершин, и *наименьшим*, если в нем наименьшее число вершин (среди всех покрытий этого графа). Число вершин в наименьшем покрытии графа G называется *числом вершинного покрытия* и обозначается через $\alpha_0(G)$.

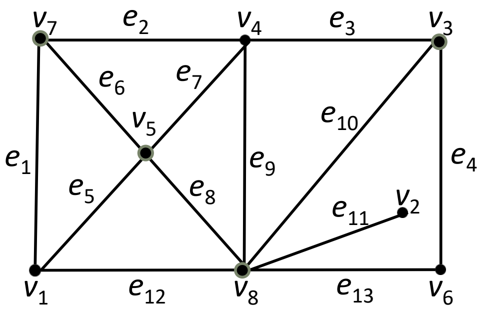


Рисунок 4.32 – Граф с выделенным наименьшим вершинным покрытием

Для некоторых графов специального вида число вершинного покрытия подсчитывается легко: для полного графа $\alpha_0(K_p) = p - 1$; для полного двудольного графа $\alpha_0(K_{p,q}) = \min(p, q)$.

Для того чтобы найти наименьшее вершинное покрытие графа G, достаточно найти покрытие столбцов его матрицы инцидентности минимальным количеством ее строк.

Матрица инцидентности графа, приведенного на рисунке 4.32, имеет вид

e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	e_{10}	e_{11}	e_{12}	e_{13}	
1	0	0	0	1	0	0	0	0	0	0	1	0	v_1
0	0	0	0	0	0	0	0	0	0	1	0	0	v_2
0	0	**1**	**1**	0	0	0	0	0	**1**	0	0	0	$\boldsymbol{v_3}$
0	1	1	0	0	0	1	0	1	0	0	0	0	v_4
0	0	0	0	**1**	**1**	**1**	**1**	0	0	0	0	0	$\boldsymbol{v_5}$
0	0	0	1	0	0	0	0	0	0	0	0	1	v_6
1	**1**	0	0	0	**1**	0	0	0	0	0	0	0	$\boldsymbol{v_7}$
0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	$\boldsymbol{v_8}$

Строки v_3, v_5, v_7 и v_8 составляют кратчайшее покрытие этой матрицы, следовательно, множество $\{v_3, v_5, v_7, v_8\}$ является наименьшим вершинным покрытием данного графа.

Понятия независимых множеств и вершинных покрытий тесно связаны. Если $G = (V, E)$ – простой граф и C – его вершинное покрытие, то множество $V \setminus C$ является независимым и наоборот, если S – независимое множество графа, то $V \setminus S$ – его вершинное покрытие. Это утверждение следует из определений этих понятий. Если S – независимое множество графа G, то оба конца ни одного ребра не содержатся в S. Это значит, что любое ребро графа имеет по крайней мере одну концевую вершину в $V \setminus S$ и одну в C (по определению, вершинного покрытия). Аналогично можно показать, что если $C_{\min}$ – наименьшее вершинное покрытие, то $V \setminus C_{\min}$ – наибольшее независимое множество и, наоборот, $C_{\min} = V \setminus S_{\max}$. Число $\beta_0(G)$ вершин в наибольшем независимом множестве $S_{\max}$

графа и число $\alpha_0(G)$ вершин в наименьшем вершинном покрытии $C_{\min}$ связаны следующим соотношением: $\alpha_0(G) + \beta_0(G) = |V|$.

Следовательно, если $\{v_3, v_5, v_7, v_8\}$ является наименьшим вершинным покрытием графа (см. рисунок 4.32), то множество $\{v_1, v_2, v_4, v_6\}$ – одно из его наибольших независимых множеств. Именно это независимое множество было найдено в пункте 4.4.2 (см. рисунок 4.29).

4.4.6 Паросочетания и реберные покрытия

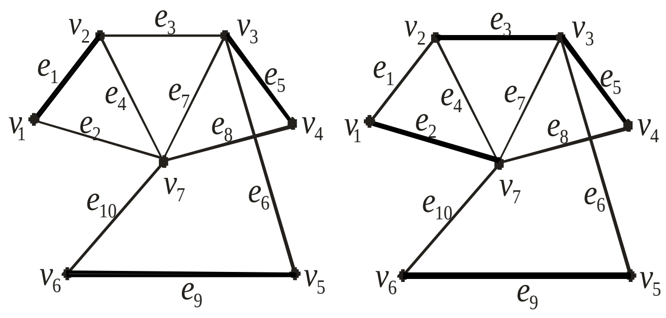
Наряду с понятием вершинной независимости важным понятием теории графов является понятие реберной независимости. Множество попарно не смежных ребер графа называется *независимым множеством ребер* или *паросочетанием*.

Задача о паросочетании в графе состоит в том, чтобы найти в нем *наибольшее паросочетание*, т. е паросочетание, имеющее наибольшее число ребер. Мощность наибольшего паросочетания называют *числом паросочетания* графа и обозначают символом $\beta_1(G)$.

С понятием паросочетания связано понятие *реберного покрытия* графа $G = (V, E)$, которым называется такое множество $E^* \subseteq E$ ребер, что всякая вершина графа покрывается хотя бы одним из этих ребер. Из этого определения следует, что реберное покрытие существует только для графов без изолированных вершин. Реберное покрытие наименьшей мощности в графе G называется его *наименьшим реберным покрытием*. Число ребер в наименьшем покрытии графа G называется *числом реберного покрытия* и обозначается через $\alpha_1(G)$.

Наибольшее паросочетание и наименьшее реберное покрытие графа на рисунке 4.33 включают ребра, изображенные полужирными линиями.

Между задачами о паросочетании и о реберном покрытии имеется тесная связь, подобная связи между задачами о независимом множестве и вершинном покрытии графа. Количественно эта связь выражается следующим образом: если E^* – наименьшее реберное покрытие, а $E_{\text{пар}}$ – наибольшее паросочетание, то $|E^*| + |E_{\text{пар}}| =$ $= |V|$ (см. рисунок 4.33).



а б

Рисунок 4.33 – Наибольшее паросочетание $\{e_1, e_5, e_9\}$ (*а*) и наименьшее реберное покрытие $\{e_2, e_3, e_5, e_9\}$ (*б*) графа

Несмотря на сходство между «вершинными» и «реберными» вариантами независимых множеств и покрытий, сложности соответствующих экстремальных задач существенно различаются: «вершинные» задачи относятся к классу труднорешаемых, а для реберных задач известны полиномиальные алгоритмы.

4.5 Раскраски и планарность

Задачи, в которых требуется найти раскраску графа или проанализировать его на планарность, имеют много приложений в различных областях человеческой деятельности. К задаче раскраски графа сводятся задачи составления разного рода расписаний, хранения и транспортировки товаров, распределения оборудования на предприятии, выбора расцветки проводов в сложных электрических схемах и многие другие практические задачи. Задача раскраски произвольных графов являлась предметом многих исследований и XIX в., и в настоящее время.

Среди проблем теории графов выделяется задача о плоской укладке графа: как изобразить граф на плоскости так, чтобы отсутствовало пересечение его ребер. К примеру, при проектировании дорог и железнодорожных путей желательно избегать наличия переездов; при изготовлении электронных микросхем печатным способом нежелательны пересечения электрических цепей.

4.5.1 Раскраска графа

Раскраской графа называется назначение цветов его вершинам (или ребрам). Пусть $G = (V, E)$ – граф, а k – натуральное число. Функция f: $V \to \{1, 2,\dots, k\}$ называется вершинной *k-раскраской*. Раскраска называется *правильной*, если $f(v_i) \neq f(v_j)$

для любых двух смежных вершин (т. е. любые две смежные вершины должны иметь разные цвета).

Задача состоит в том, чтобы раскрасить вершины графа G в минимальное число цветов. Это число называется *хроматическим числом* графа и обозначается через $\chi(G)$. Если $\chi(G) = k$, то граф называется *k-хроматическим*. Хроматические числа некоторых графов находятся просто: $\chi(G) = n$ для полного графа K_n, для пустого графа $\chi(G) = 1$, для двудольного графа $\chi(G) = 2$.

Из определения раскраски графа следует, что хроматическое число графа не может быть меньше числа $\varphi(G)$ вершин его наибольшей клики, и это число является нижней оценкой $\chi(G) \geq \varphi(G)$ хроматического числа этого графа.

Соответствующая хроматическому числу $\chi(G) = k$ раскраска вершин разбивает множество вершин на k подмножеств $V_1, V_2, \ldots, V_k$, каждое из которых содержит вершины одного цвета. Эти множества являются независимыми, так как в каждом из них нет смежных вершин. Число вершин каждого из цветных классов не превышает $\beta_0(G)$ – числа независимости графа G. Отсюда следует, что для графа с n вершинами $\chi(G)\times\beta_0(G) \geq n$. Нетрудно убедиться, что это неравенство выполняется для упомянутых полного, пустого и двудольного графов. Из полученного неравенства получается еще одна нижняя оценка хроматического числа графа: $\chi(G) \geq \dfrac{n}{\beta_0(G)}$.

Зная все максимальные независимые множества графа, минимальную раскраску вершин можно получить следующим образом: найти кратчайшее покрытие множества вершин графа максимальными независимыми множествами; а затем удалить некоторые вершины из элементов полученного покрытия, добившись того, чтобы каждая вершина входила ровно в одно из выделенных независимых множеств.

Найдем минимальную раскраску вершин графа $G = (V, E)$ (рисунки 4.32, 4.34). Для этого графа найдено (пункт 4.4.2) шесть максимальных независимых множеств:

$$\{v_2, v_3, v_7\}, \{v_7, v_8\}, \{v_2, v_6, v_7\}, \{v_2, v_3, v_5\}, \{v_2, v_5, v_6\}, \{v_1, v_2, v_4, v_6\}.$$

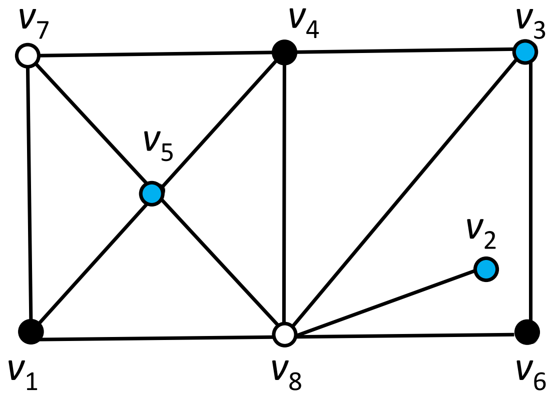


Рисунок 4.34 – Раскраска вершин графа

Зададим это множество в матричном виде:

v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	
1	1	1	0	0	0	0	0	$\{v_1, v_2, v_3\}$
0	0	0	0	0	0	**1**	**1**	$\boldsymbol{\{v_7, v_8\}}$
0	1	0	0	0	1	1	0	$\{v_2, v_6, v_7\}$
0	**1**	**1**	0	**1**	0	0	0	$\boldsymbol{\{v_2, v_3, v_5\}}$
0	1	0	0	1	1	0	0	$\{v_2, v_5, v_6\}$
1	**1**	0	**1**	0	**1**	0	0	$\boldsymbol{\{v_1, v_2, v_4, v_6\}}$

Кратчайшее покрытие столбцов этой матрицы включает три строки, соответствующие максимальным независимым множествам $A = \{v_7, v_8\}$, $B = \{v_2, v_3, v_5\}$ и $C = \{v_1, v_2, v_4, v_6\}$. Перейдем от покрытия множества V вершин ($A \cup B \cup C = V$, но $B \cap C \neq \varnothing$) к одному из его разбиений, задающему раскраску вершин графа в три цвета: $A = \{v_7, v_8\}$, $B = \{v_2, v_3, v_5\}$ и $C = \{v_1, v_4, v_6\}$ (см. рисунок 4.34).

Для графов, число независимых множеств которых невелико, приведенный способ раскраски вершин является приемлемым. Однако число максимальных независимых множеств для некоторых графов может оказаться настолько большим, что данный способ вообще не сможет быть практически реализован. Существует немало методов раскраски, не использующих задачу покрытия и получающих точно минимальное число цветов, но их применение существенно ограничено размерностью задачи.

Рассмотрим эвристический метод раскраски графа, который не гарантирует получения минимума цветов, но позволяет получить раскраску, близкую к минимальной. Процесс раскраски графа $G = (V, E)$ представляет собой последовательность шагов, на каждом из которых выбирается вершина и окрашивается в определенный цвет. Текущая ситуация характеризуется следующими объектами:

k – число задействованных цветов;

Y – множество еще не раскрашенных вершин;

$B_1, B_2, \dots, B_k$ – совокупность подмножеств множества вершин V, такая, что B_i ($i = 1, 2, \dots, k$) содержит те и только те вершины из множества Y, которые нельзя раскрасить в i-й цвет.

При раскраске вершин графа будем следовать следующим правилам:

1) правило введения новой краски: новый цвет вводится для вершины, которая не может быть раскрашена ни в один из имеющихся цветов;

2) правило сохранения возможностей: если в цвет C_i можно раскрасить вершины, составляющие множество V_i, то нераскрашенная вершина $v \in V_i$ красится в этот цвет, если после этого сохраняется возможность окраски в этот цвет любой другой вершины из V_i;

3) правило выбора с риском: выбирается нераскрашенная вершина v и цвет C_i, такие, что после раскраски v в цвет C_i возможность раскраски в цвет C_i теряет минимальное число нераскрашенных вершин.

Шаги алгоритма раскраски, на которых применяются первые два правила, являются «хорошими», а шаги, на которых используется третье правило – «сомнительными». Если в процессе раскраски выполняются только «хорошие» шаги, то получаемая раскраска будет минимальной. Если же приходится выполнять также и «сомнительные» шаги, то полученная раскраска может оказаться не минимальной. Если задаться целью получить минимальную раскраску, то необходимо в сомнительных случаях производить перебор вариантов выбора пары «вершина, цвет».

Рассмотрим случаи, когда при раскраске вершин графа появляется возможность выполнения «хорошего» шага и когда приходится выполнять «сомнительный» шаг.

1. Имеется вершина $v \in Y$, такая, что $v \in B_i$ для всех $i = 1, 2, \dots, k$.

В этом случае применяется правило введения новой краски: вершина v красится в цвет C_{k+1} и удаляется из множества Y и из всех множеств B_i, где она была. Формируется множество B_{k+1} (окрестность $\Gamma' v$ на множестве Y вершин) и число k красок увеличивается на единицу. Если же таких вершин $v \in Y$ несколько, то из них выбирается та, для которой ее множество B_{k+1} имеет максимальную мощность.

2. Имеется вершина $v \in Y$ и цвет C_i, такие, что $v \notin B_i$ и $\Gamma v \cap Y \subseteq B_i$, что означает, что все смежные с v нераскрашенные вершины не могут быть раскрашены в этот цвет i.

В этом случае возможно применение правила сохранения возможностей: вершина v красит в цвет C_i и удаляется из множества Y и из всех множеств B_j, где она была.

3. Все вершины из Y могут быть раскрашены в один из k цветов, но для любой вершины $v \in Y$ находится только такой цвет C_i, что $v \notin B_i$, но не выполняется условие $\Gamma v \cap Y \subseteq B_i$, что означает, что после раскраски вершины v в цвет C_i в множество B_i добавятся новые вершины.

В этом случае приходится применять правило выбора с риском: выбираются вершина $v \in Y$ и цвет C_i, такие, что $v \notin B_i$ и приращение $\Delta |B_i|$ мощности множества B_i минимально среди всех остальных возможностей выбора пар «вершина v, цвет C_i» ($v \in Y$, $i = 1, 2, \dots, k$). Вершина v удаляется из Y и из всех B_j, где она была, и красится в цвет C_i.

Выполнение описанных действий повторяется до тех пор, пока множество Y не станет пустым.

Продемонстрируем выполнение приведенного метода для графа, изображенного на рисунке 4.35. В начальной ситуации $Y = V$, $k = 0$, выбирается вершина с максимальной степенью и красится в цвет C_1. При выполнении дальнейшей раскраски вершин графа реализуется следующая последовательность шагов, для каждого из которых указывается применяемое правило, цветные классы, множества B_i и Y.

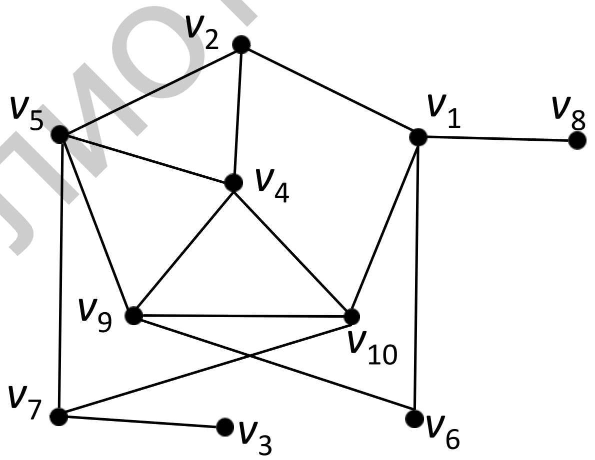


Рисунок 4.35 – Пример графа

1. $\{\{v_1\}\}$; $B_1 = \{v_2, v_6, v_8, v_{10}\}$; $Y = \{v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}\}$.

2. (Правило 1: $v_{10} \in B_1$ и $\Gamma' v_{10} = \{v_4, v_7, v_9\} = \max$): $\{\{v_1\}, \{v_{10}\}\}$; $B_1 = \{v_2, v_6, v_8\}$, $B_2 = \{v_4, v_7, v_9\}$; $Y = \{v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$.

3. (Правило 2: $v_8 \notin B_2$ и $\Gamma' v_8 = \varnothing$): $\{\{v_1\}, \{v_{10}, v_8\}$; $B_1 = \{v_2, v_6\}$, $B_2 = \{v_4, v_7, v_9\}$; $Y = \{v_2, v_3, v_4, v_5, v_6, v_7, v_9\}$.

4. (Правило 3: $v_3 \notin B_1$ и $\Delta|B_1| = 1 = \min$): $\{\{v_1, v_3\}, \{v_8, v_{10}\}\}$; $B_1 = \{v_2, v_6, v_7\}$, $B_2 = \{v_4, v_7, v_9\}$; $Y = \{v_2, v_4, v_5, v_6, v_7, v_9\}$.

5. (Правило 1: $v_7 \in B_1 \cap B_2$): $\{\{v_1, v_3\}, \{v_8, v_{10}\}, \{v_7\}\}$; $B_1 = \{v_2, v_6\}$, $B_2 = \{v_4, v_9\}$; $B_3 = \{v_5\}$; $Y = \{v_2, v_4, v_5, v_6, v_9\}\}$.

6. (Правило 3: $v_5 \notin B_2$ и $\Delta|B_2| = 1 = \min$): $\{\{v_1, v_3\}, \{v_5, v_8, v_{10}\}, \{v_7\}\}$; $B_1 = \{v_2, v_6\}$, $B_2 = \{v_2, v_4, v_9\}$; $B_3 = \{\ \}$; $Y = \{v_2, v_4, v_6, v_9\}$.

7. (Правило 2: $v_6 \notin B_2$ и $\Delta|B_6| = 0$): $\{\{v_1, v_3\}, \{v_5, v_6, v_8, v_{10}\}, \{v_7\}\}$; $B_1 = \{v_2\}$, $B_2 = \{v_2, v_4, v_9\}$; $B_3 = \{\ \}$; $Y = \{v_2, v_4, v_9\}$.

8. (Правило 3: $v_2 \notin B_3$ и $\Delta|B_3| = 1$): $\{\{v_1, v_3\}, \{v_5, v_6, v_8, v_{10}\}, \{v_2, v_7\}\}$; $B_1 = \{\ \}$, $B_2 = \{v_4, v_9\}$; $B_3 = \{v_4\}$; $Y = \{v_4, v_9\}$.

9. (Правило 2: $v_9 \notin B_3$ и $\Delta|B_3| = 0$): $\{\{v_1, v_3\}, \{v_5, v_6, v_8, v_{10}\}, \{v_2, v_7, v_9\}\}$; $B_1 = \{\ \}$, $B_2 = \{v_4\}$; $B_3 = \{v_4\}$; $Y = \{v_4\}$.

10. (Правило 2: $v_4 \notin B_1$ и $\Delta|B_1| = 0$): $\{\{v_1, v_3, v_4\}, \{v_5, v_6, v_8, v_{10}\}, \{v_2, v_7, v_9\}\}$; $Y = \{\ \}$.

Работа алгоритма завершается с результатом $\{\{v_1, v_4, v_7\}, \{v_2, v_3, v_8, v_9\}, \{v_5, v_6, v_{10}\}\}$. Полученная раскраска данного графа является минимальной, так как хроматическое число графа не может быть меньше кликового числа графа, а оно равно трем, так как на рисунке 4.35 нетрудно обнаружить клики, состоящие из трех вершин.

Иногда можно получить раскраску графа, минимальную или близкую к минимальной, с помощью так называемого «*жадного*» алгоритма, где на каждом шаге в текущий цвет раскрашивается как можно больше вершин. Желательно для этого каждый раз брать вершины из наибольшего независимого множества. Раскрашенные вершины удаляются из графа, затем вводится новый цвет, в него опять раскрашивается как можно больше вершин и так далее до тех пор, пока множество вершин графа не станет пустым.

Для графа на рисунке 4.35 «*жадный*» алгоритм выберет сначала одно из наибольших независимых множеств, например $\{v_2, v_3, v_6, v_8, v_{10}\}$, раскрасив его вершины в цвет 1, затем наибольшее независимое множество (на множестве оставшихся вершин), например $\{v_1, v_4, v_7\}$, раскрасив его вершины в цвет 2. После этого остаются две смежные вершины v_5 и v_9, соответственно для раскраски графа потребуется 4 краски. Однако если выбрать в качестве начального наибольшего независимого множества $\{v_3, v_5, v_6, v_8, v_{10}\}$, то дальше будут выбраны $\{v_1, v_4, v_7\}$ и $\{v_2, v_9\}$, породив трехцветную раскраску графа.

Наряду с рассмотренной задачей раскраски вершин иногда ставится задача раскраски ребер графа $G = (V, E)$. Для этой задачи введенные выше определения правильной и минимальной раскраски для случая вершин аналогично перефразируются и для случая ребер. Задача правильной реберной раскраски состоит в разбиении множества ребер E на непересекающиеся паросочетания $E_1, E_2, \ldots, E_p$ (состоящие из попарно не смежных ребер), каждое из которых состоит из ребер, раскрашиваемых в один цвет.

Задачу реберной раскраски можно свести также к раскраске вершин, если перейти от графа $G = (V, E)$ к его *реберному графу* $L(G)$ (рисунок 4.36). Вершины графа $L(G)$ соответствуют ребрам графа G, и две вершины графа $L(G)$ связаны ребром, если и только если соответствующие им ребра графа G имеют общую инцидентную им вершину в G. Раскраска ребер графа G соответствует раскраске вершин графа $L(G)$.

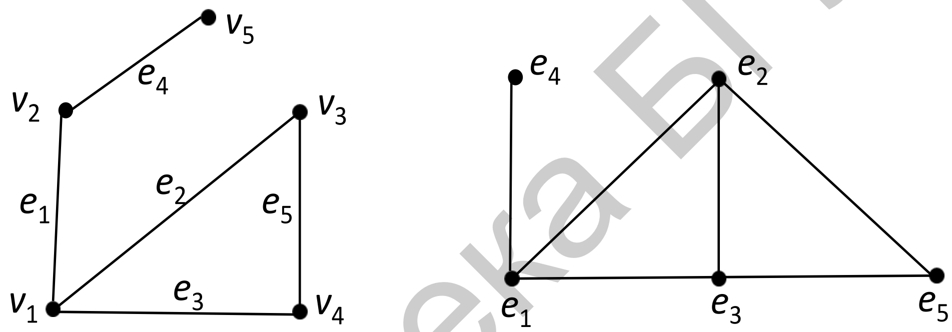


Рисунок 4.36 – Граф $G = (V, E)$ и его реберный граф $L(G)$

4.5.2 Бихроматические графы

Граф G называется *k-хроматическим*, если его хроматическое число $\chi(G) = k$. Очевидно, что только пустые графы являются 1 – хроматическими. Особый класс k-хроматических графов составляют *бихроматические графы*, т. е. такие, у которых $\chi(G) = 2$ (рисунок 4.37). Необходимое и достаточное условие бихроматичности графа устанавливается теоремой Кенига.

Теорема Кенига. Непустой граф является бихроматическим тогда и только тогда, когда он не содержит циклов нечетной длины.

Очевидно, что граф, имеющий цикл нечетной длины не может быть раскрашен в два цвета, так как среди вершин, число которых нечетно, обязательно окажутся одноцветные. Пусть $G = (V, E)$ – связный бихроматический граф, что не нарушает общности, так как в случае несвязного графа последующие рассуждения можно провести для каждой его компоненты связности в отдельности. Обозначим через V^1 и V^2 множества вершин графа G, раскрашенных соответственно в цвета 1 и 2. Всякое ребро соединяет вершину из V^1 с вершиной из V^2. Следовательно, всякая цепь, начинающаяся и оканчивающаяся в одном и том же множестве V^i (i = 1, 2), имеет четную длину, т. е. граф не имеет циклов нечетной длины.

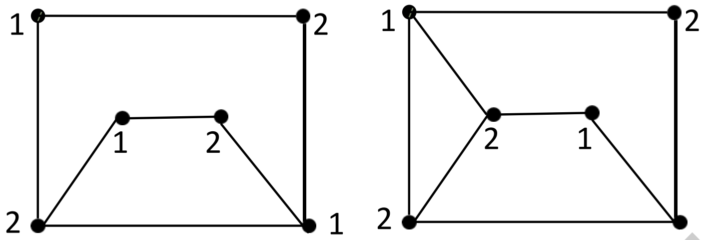


Рисунок 4.37 – Бихроматический и небихроматический графы

Пусть теперь связный граф G не имеет циклов нечетной длины. Возьмем любую вершину v из графа G. Сформируем множество V^1 из вершин, отстоящих от v на четном расстоянии в графе G, и множество V^2 из всех остальных вершин. Ни одна пара вершин из V^2 не связана ребром. Действительно, если бы имелось ребро $(v_i, v_j) \in E$, у которого $v_i \in V^2$ и $v_j \in V^2$, то в графе имелся бы цикл, составленный из этого ребра и, во-первых, цепи, связывающей v_i и v_j и не проходящей через v, если такая имеется (а она может иметь только четную длину при принятом предположении об отсутствии циклов нечетной длины), или, во-вторых, цепи, соединяющей v с v_i, и цепи, соединяющей v_j с v – в противном случае. В обоих случаях получался бы цикл нечетной длины, что противоречит условию.

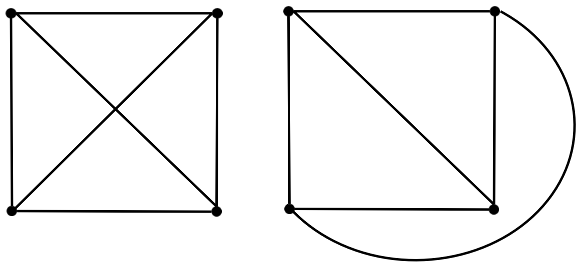
Очевидно, всякий двудольный граф, имеющий хотя бы одно ребро, является бихроматическим, так как любая доля двудольного графа представляет собой независимое множество.

Бихроматичность графа легко установить, используя способ последовательной раскраски (см. рисунок 4.37). Для этого произвольно выбирается вершина v и красится в цвет 1. Вершины ее окрестности Γv красятся в цвет 2, неокрашенные вершины из окрестностей вершин, принадлежащих Γv, красятся в цвет 1 и т. д. В результате либо граф раскрашивается в два цвета, либо на каком-то шаге смежные вершины оказываются окрашенными в один и тот же цвет. Это говорит о том, что граф не является бихроматическим.

4.5.3 Планарные графы

Граф *укладывается* на плоскости, если его можно так нарисовать на ней, что никакие два ребра не будут пересекаться, т. е. не будут иметь общих точек, кроме инцидентной им вершины. *Плоский* граф – это граф, уложенный на плоскости.

Граф называется *планарным*, если его можно уложить на плоскости. Получаемый при этом плоский граф называют также плоской *укладкой* планарного графа. На рисунке 4.38 приведен планарный граф и его плоские укладки.

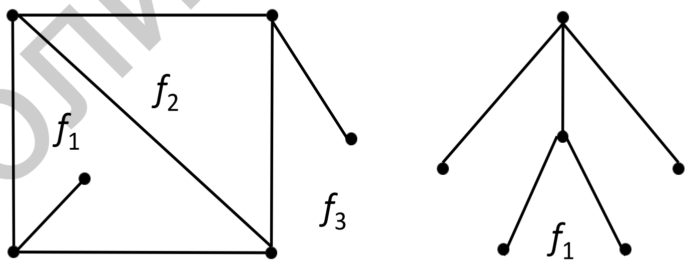


а *б* *в*

Рисунок 4.38 – Планарный граф (*а*) и его плоские укладки (*б*, *в*)

Гранью плоского графа называется максимальная по включению область плоскости, ограниченная ребрами, любые две точки которой могут быть соединены непрерывной линией, не пересекающей ребра графа. Каждый плоский граф имеет единственную неограниченную грань, которая называется *внешней*. Все остальные его грани называются *внутренними*.

Плоский граф, изображенный на рисунке 4.39, *а*, имеет три грани: f_1 и f_2 – внутренние грани, f_3 – внешнюю, а граф на рисунке 4.39, *б* имеет только одну



грань – внешнюю.

а *б*

Рисунок 4.39 – Грани плоских графов

Теорема Эйлера о числе граней (формула Эйлера). Для всякого связного плоского (n, m)-графа, имеющего f граней, справедливо равенство

$$n - m + f = 2.$$

Действительно, для дерева имеем $m = n - 1$ и $f = 1$ (см. рисунок 4.39, *б*). Следовательно, данная формула для дерева верна. Увеличивая число ребер в графе на некоторую величину при сохранении числа вершин, мы увеличиваем число граней на ту же величину.

Из теоремы Эйлера следует, что число f граней любой плоской укладки связного планарного графа постоянно и равно $f = m - n + 2$, что говорит о том, что число граней является инвариантом планарного графа.

Формула Эйлера легко обобщается на случай несвязного графа: $n - m + f = 1 + c$, где c – число компонент связности в графе.

Из теоремы Эйлера следует, что для всякого планарного связного (n, m)-графа с числом вершин $n \geq 3$ число ребер удовлетворяет неравенству

$$m \leq 3n - 6. \tag{4.4}$$

Неравенство выводится из следующих рассуждений. Так как каждая грань ограничена по меньшей мере тремя ребрами (так как граф не имеет кратных ребер и петель), а каждое ребро ограничивает не более двух граней, то $f \leq \frac{2m}{3}$. Подставляя значение в формулу Эйлера, получаем искомый результат.

Утверждение. Граф K_5 и полный двудольный граф $K_{3,3}$ (рисунок 4.40) являются непланарными графами.

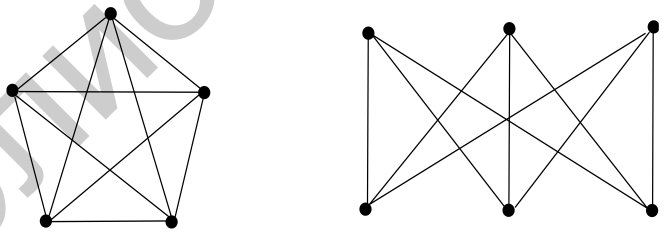


Рисунок 4.40 – Простейшие непланарные графы K_5 и $K_{3,3}$

Утверждение доказывается от противного. Действительно, для графа K_5 $n = 5$, $m = 10$, поэтому неравенство (4.4) для этого графа не выполняется: $m = 10 \leq 9$. Значит, предположение о планарности графа не верно.

Для графа $K_{3,3}$ имеем $n = 6$, $m = 9$ и из формулы Эйлера следует, что $f = 5$. В то же время в графе $K_{3,3}$ нет граней-треугольников, поэтому если предположить, что $K_{3,3}$ планарен, то каждая его грань должна быть ограничена не менее чем четырьмя ребрами и $4f \leq 2m$, последнее приводит к противоречию $20 \leq 18$.

Графы K_5 и $K_{3,3}$ являются простейшими непланарными графами. Нетрудно убедиться, что удаление любой вершины или любого ребра превращает их в планарные графы (рисунок 4.41). Эти два графа используются при анализе графов на планарность и известны как графы Куратовского.

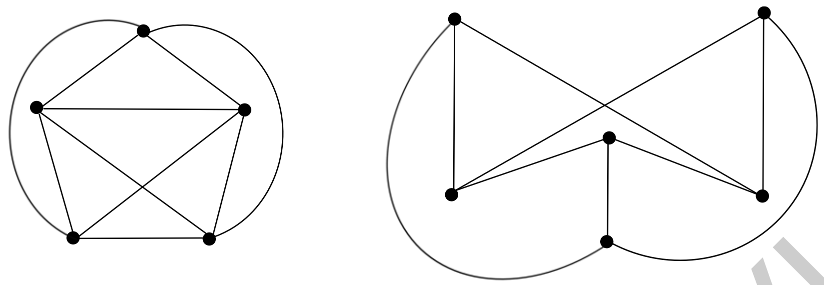


Рисунок 4.41 – Попытка укладывания на плоскость непланарных графов K_5 и $K_{3,3}$

С привлечением графа K_5 решается занимательная задача о пяти соседях: можно ли проложить дорожки между каждой парой домов пяти соседей так, чтобы они не пересекались? С привлечением $K_{3,3}$ решается занимательная задача о трех домах и трех колодцах: имеются три дома и три колодца неподалеку от них. Можно ли проложить от каждого дома дорожки ко всем трем колодцам так, чтобы эти дорожки не пересекались? Приводимая ниже теорема Понтрягина – Куратовского дает отрицательный ответ на эти вопросы.

Введем некоторые определения, необходимые для формулировки теоремы Понтрягина – Куратовского.

Операция подразбиения ребра (u, v) в графе $G = (V, E)$ $(u, v \in V)$ состоит в добавлении в множество V новой вершины w и в замене ребра (u, v) парой новых последовательно соединенных ребер (u, w) и (w, v) (рисунок 4.42).

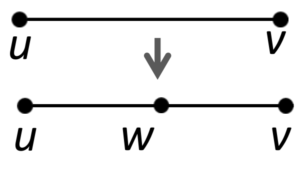


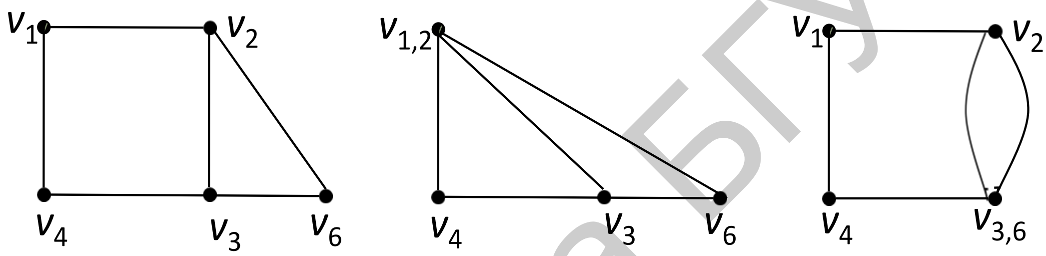
Рисунок 4.42 – Подразбиение ребра (u, v)

Два графа называются *гомеоморфными*, если их можно получить из одного и того же графа путем последовательных подразбиений ребер.

Теорема Понтрягина – Куратовского. Граф планарен тогда и только тогда, когда он не содержит подграфов, гомеоморфных графам K_5 или $K_{3,3}$.

Достаточно сложное доказательство этой теоремы здесь не приводится. Кроме классического результата Понтрягина и Куратовского были предложены и другие критерии планарности графа. Приведем еще один критерий, основанный на понятии стягивания ребер.

Операция *стягивания* ребра (u, v) в графе $G = (V, E)$ состоит в удалении этого ребра и отождествлении вершин, смежных u и v, т. е. замене их новой вершиной, которая будет инцидентна тем ребрам графа, которым инцидентны вершины u и v (рисунок 4.43). При выполнении операции стягивания в общем случае могут появиться кратные ребра.



а *б* *в*

Рисунок 4.43 – Граф $G = (V,E)$ (*а*) и графы, полученные из него стягиванием ребра v_1v_2 (*б*) и ребра v_3v_6 (*в*)

Граф G называется *стягиваемым* к графу H, если H можно получить из G последовательностью операций стягивания ребер. Нетрудно заметить, что граф Петерсена (рисунок 4.44) стягиваем к полному графу K_5.

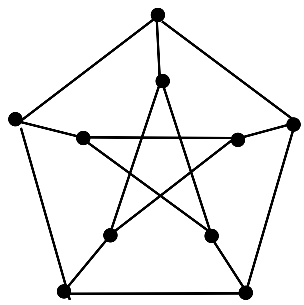


Рисунок 4.44 – Граф Петерсена, стягиваемый к графу K_5

Теорема Вагнера. Граф планарен тогда и только тогда, когда он не содержит подграфов, стягиваемых к графам K_5 или $K_{3,3}$.

Для непланарных графов вводятся характеристики, отражающие меру их непланарности. *Искаженностью* графа или числом планарности называется наименьшее число его ребер, удаление которых делает граф планарным. Нетрудно заметить, что искаженность графов K_5 и $K_{3,3}$ равна 1 (см. рисунок 4.41).

4.5.4 Раскраска планарных графов

Проблема раскраски планарных графов является одной из самых известных проблем в теории графов. Возникла она в середине XIX в. Первоначально эта проблема формулировалась в следующем виде: достаточно ли четырех красок для такой раскраски географической карты, при которой любые две смежные страны не были бы окрашены в один цвет. При этом рассматривались такие карты, где каждая страна представлена связной областью, т. е. граница любой страны состоит из одной замкнутой линии. Смежными называются страны, которые имеют общую границу ненулевой длины, т. е. границу в виде линии (не одной общей точки).

Позднее понятие карты и задача ее раскраски были формализованы. *Картой* назван связный плоский мультиграф без мостов. В 1879 г. британский математик А. Кэли сформулировал гипотезу четырех красок: всякая карта может быть раскрашена в четыре цвета.

Сформулируем задачу раскраски карт в терминах теории графов. Построим для плоского мультиграфа G, задающего карту, граф G^*, *геометрически двойственный* к G. Будем считать грани мультиграфа G, имеющие общее ребро, смежными.

Граф G^*, геометрически двойственный по отношению к плоскому графу G, строится следующим образом. Вершины графа G^* ставятся в соответствие граням графа G. Каждому ребру e графа G соотносится ребро графа G^*, которое соединяет его вершины, соответствующие граням, имеющим общее ребро e в графе G. Пример построения двойственного графа G^* приведен на рисунке 4.45, где его ребра показаны тонкими кривыми линиями.

Раскраска граней плоского графа, при которой соседние грани раскрашиваются в различные цвета, эквивалентна раскраске вершин его двойственного графа. Таким образом, задача раскраски географической карты сводится к раскраске вершин графа минимальным числом цветов.

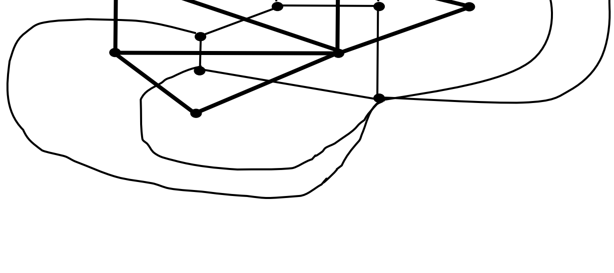


Рисунок 4.45 – Плоский граф и граф, двойственный ему

Доказано, что всякий планарный граф можно раскрасить не более чем в пять цветов, и долгое время гипотеза четырех красок, оставалась только гипотезой. Было много безуспешных попыток ее доказать. В 1976 г. появилось сообщение о том, что данная гипотеза доказана с помощью вычислительной машины. Для этого ранее были найдены 1 482 конфигурации и дано доказательство утверждения, что если в максимальном планарном графе подграф вида такой конфигурации раскрашивается в четыре цвета, то и сам граф раскрашивается в четыре цвета. С помощью мощной вычислительной машины все графы, изоморфные таким конфигурациям, были раскрашены в четыре цвета приблизительно за 2 000 ч машинного времени. Однако в связи со сложностью проверки упомянутых доказательств не все специалисты с ними согласны.

4.6 Циклы и разрезы графа

4.6.1 Деревья, леса, остовы

Самым простым в некотором смысле типом графов являются деревья. Графы этого специального вида выделяются особо, потому что они часто используются в различных вычислительных приложениях, например в программировании.

Граф без циклов называется *ациклическим* графом. Связный ациклический граф называется *деревом*. Обобщением дерева является *лес* – несвязный граф, компонентами связности которого являются деревья.

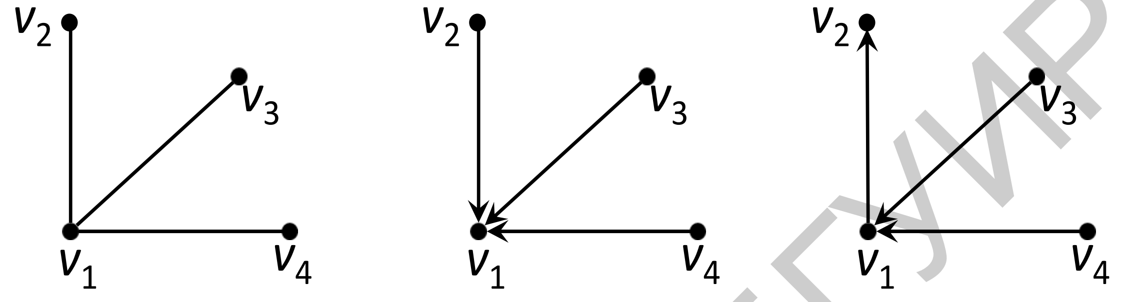
Пусть $G = (V, E)$ – граф с n вершинами и m ребрами. Следующие утверждения дают эквивалентные определения дерева:

1. Дерево – связный ациклический граф.
2. Дерево – связный граф, число ребер m которого: $m = n - 1$.
3. Дерево – ациклический граф, число ребер которого $m = n - 1$.
4. Дерево – это граф, в котором каждая пара вершин связана одной и только одной простой цепью.
5. Дерево – ациклический граф, такой, что при соединении ребром произвольных двух несмежных его вершин получается граф, имеющий ровно 1 цикл.

Из этих определений следует, что для леса как обобщения дерева справедливо следующее утверждение.

Утверждение. Число ребер m леса с k компонентами связности: $m = n - k$.

Вершина дерева является *концевой* или *висячей*, если ее степень равна 1. Пусть в дереве отмечена некоторая вершина v_0. Эта вершина называется *корнем* дерева, а само дерево – деревом с корнем. В таком дереве можно естественным образом ориентировать ребра так, что любую вершину можно соединить путем с корнем. Полученное таким образом дерево с корнем называется *ориентированным* деревом (рисунок 4.46). Корень ориентированного дерева – единственная вершина, которая имеет нулевую степень исхода.



а *б* *в*

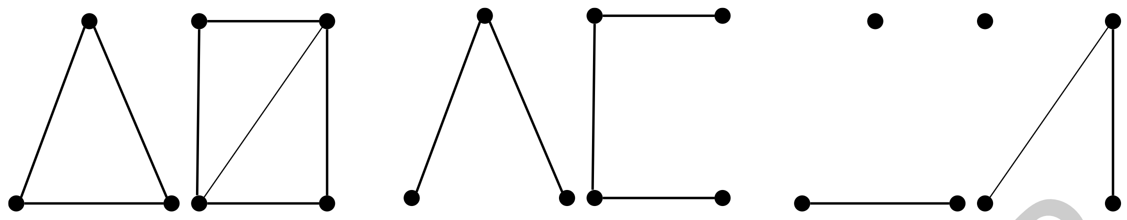
Рисунок 4.46 – Деревья без корня (*а*), с корнем v_1 (*б*) и с корнем v_2 (*в*)

Всякий связный неориентированный (n, m)-граф G имеет остовный подграф в виде дерева, называемый *остовным деревом* или *остовом*. Остовное дерево представляет минимальное множество ребер графа, связывающее все его вершины. Число ребер в остовном дереве равно $n - 1$, в остовном лесе – $n - k$.

Остовный подграф графа G, содержащий только те ребра, которые не входят остов, называется *ко-деревом*. Всякий остов однозначно определяет свое ко-дерево или *ко-лес* (рисунок 4.47). Остов по определению имеет $n - k$ ребер, а ко-лес $m - n + k$ ребер.

4.6.2 Базис циклов. Цикломатическое число графа

Рассмотрим остовное дерево $T = (V, D)$ некоторого связного графа $G = (V, E)$ $(D \subseteq E)$ (рисунок 4.48). Добавление в остов T одного ребра l_i из множества $L = E \setminus D$ ребер его ко-дерева $K = (V, L)$ приводит к появлению в графе $T \cup \{l_i\}$ точно одного простого цикла. Этот цикл состоит из добавленного ребра и тех ребер дерева T, которые принадлежат единственной цепи, соединяющей в T концы данного ребра.



а *б* *в*

а – граф *G* с двумя компонентами связности;
б – остов *T* (остовный лес) графа *G*; *в* – ко-лес остова *T*

Рисунок 4.47 – Граф и его подграфы

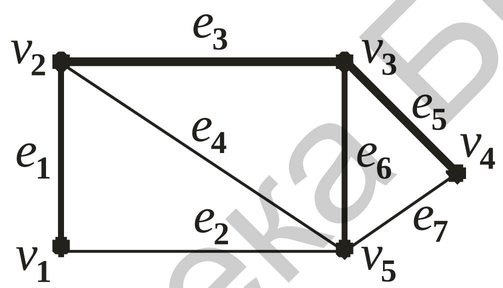


Рисунок 4.48 – Граф с выделенным остовным деревом

Каждый из циклов графа, получаемых путем добавления одного ребра из множества $E \setminus D$ в остовный граф *T*, имеет ребро, не принадлежащее никакому другому из этих циклов. В этом смысле эти циклы независимы. Множество циклов, определяемых с помощью остовного дерева, называется *базисом циклов* графа *G*, а сами циклы, принадлежащие базису, – *фундаментальными* (или базисными) циклами.

Всякий цикл (*n*, *m*)-графа *G* можно представить *m*-мерным двоичным вектором, в котором *i*-я компонента имеет значение 1 или 0 в зависимости от того, принадлежит или нет *i*-е ребро данному циклу. В векторе, представляющем каждый фундаментальный цикл базиса, имеется компонента, имеющая в нем значение 1 и значение 0 в векторах других циклов базиса. Любой небазисный цикл можно выразить в виде линейной комбинации фундаментальных циклов. Это означает, что задающий его вектор **c** выражается в виде покомпонентной суммы по модулю 2 векторов $\mathbf{c}_i$, представляющих фундаментальные циклы:

$$\mathbf{c} = a_1\mathbf{c}_1 \oplus a_2\mathbf{c}_2 \oplus \ldots \oplus a_r\mathbf{c}_r,$$

где $a_i \in \{0, 1\}$.

Число фундаментальных циклов в графе T (и в графе G) равно числу ребер ко-дерева: $m - n + 1$. В общем случае, когда граф G k-связен, это число равно $\nu(G) = m - n + k$ и называется *цикломатическим числом* графа G. Число $\rho(G) = n - k$ ребер в остове называется *коцикломатическим числом*.

Сумма цикломатического и коцикломатического чисел графа равна числу m ребер графа G:

$$\nu(G) + \rho(G) = m - n + k + n - k = m.$$

Заметим, что базис циклов определяется неоднозначно и состав его зависит от выбранного остовного дерева.

Любой цикл, не принадлежащий базису, может быть выражен в виде линейной комбинации фундаментальных циклов. Покажем, как можно выразить произвольный цикл графа через базисные циклы.

Для иллюстрации процесса построения базиса циклов рассмотрим связный граф, приведенный на рисунке 4.48, в котором выделенные ребра образуют остовное дерево. Остальные ребра принадлежат ко-дереву, которое определяет три фундаментальных цикла:

– цикл 1 $\{e_1, e_3, e_6, e_2\}$, порождаемый ребром e_2 и проходящий через вершины v_1, v_2, v_3, v_5;

– цикл 2 $\{e_3, e_6, e_4\}$, порождаемый ребром e_4 и проходящий через вершины v_2, v_3, v_5;

– цикл 3 $\{e_6, e_5, e_7\}$, порождаемый ребром e_7 и проходящий через вершины v_3, v_4, v_5.

Возьмем два последних цикла. Их линейная комбинация представит цикл $\{e_3, e_5, e_7, e_4\}$, проходящий через вершины v_2, v_3, v_4, v_5 и не являющийся фундаментальным:

$$\begin{array}{ccccccc}
e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 \\
\hline
0 & 0 & 1 & 1 & 1 & 0 & 1
\end{array}$$

Следует отметить, что цикл, представляющий собой линейную комбинацию фундаментальных циклов, не обязательно является простым. Например, линейная комбинация всех трех фундаментальных циклов графа, приведенного на рисунке 4.48, является циклом, который проходит два раза через вершину v_5.

e_1	e_2	e_3	e_4	e_5	e_6	e_7
1	1	1	0	0	1	0
0	0	1	1	0	1	0
0	0	0	0	1	1	1
1	1	0	1	1	1	1

Кроме того, результат линейной комбинации фундаментальных циклов не всегда порождает цикл. Например, граф, приведенный на рисунке 4.49, имеет три фундаментальных цикла, порождаемых выделенным на нем остовным деревом:

– цикл 1 $\{e_1, e_2, e_3, e_4\}$, порождаемый ребром e_1 и проходящий через вершины v_1, v_2, v_3, v_4;

– цикл 2 $\{e_3, e_4, e_5\}$, порождаемый ребром e_5 и проходящий через вершины v_1, v_3, v_4;

– цикл 3 $\{e_7, e_8, e_9\}$, порождаемый ребром e_9 и проходящий через вершины v_5, v_6, v_7.

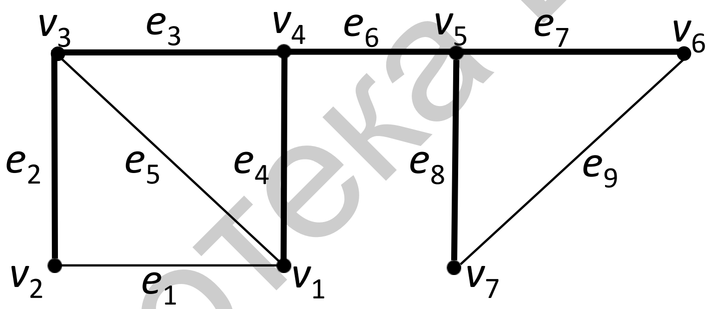


Рисунок 4.49 – Граф с выделенным остовным деревом

Линейная комбинация всех фундаментальных циклов этого графа не является циклом:

e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
1	1	1	1	0	0	0	0	0
0	0	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1
1	1	0	0	1	0	1	1	1

4.6.3 Базис разрезов

Разрезом графа называется множество его ребер, удаление которых увеличивает число компонент связности графа. Имеет смысл говорить только о минимальных разрезах, т. е. о таких разрезах, каждый из которых перестает быть разрезом при удалении из него любого ребра. Под разрезом далее понимается именно минимальный разрез.

Разрез графа связан с его остовным деревом. Остовное дерево представляет минимальное множество ребер графа, связывающее все его вершины, а разрез – минимальное множество ребер, разделяющее вершины. Очевидно, любое остовное дерево графа должно иметь хотя бы одно общее ребро с каждым разрезом.

Пусть в связном графе G, число вершин которого n, выделено остовное дерево $T = (V, D)$. Назовем *фундаментальным разрезом* каждый из $n – 1$ разрезов, который содержит одно и только одно ребро, принадлежащее дереву T. Таким образом, каждое ребро e дерева T определяет фундаментальный разрез, который составляют, кроме ребра e, все те ребра ко-дерева $K = (V, L)$ остова T, которые не принадлежат дереву T, но входят в фундаментальные циклы, содержащие выделенное ребро e. Действительно, чтобы разделить вершины, связанные ребром e, надо удалить ребро e и разорвать все цепи, связывающие данные вершины помимо ребра e. А каждая такая цепь вместе с ребром e образует фундаментальный цикл.

Множество фундаментальных разрезов графа G называется *базисом разрезов* графа G. Любой разрез графа G представляется линейной комбинацией его фундаментальных разрезов.

Для графа, приведенного на рисунке 4.48, множества ребер $\{e_3, e_2, e_4\}$ и $\{e_6, e_2, e_4, e_7\}$ представляют собой фундаментальные разрезы. Линейная комбинация этих разрезов порождает разрез $\{e_3, e_6, e_7\}$:

$$
\begin{array}{ccccccc}
e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \\
0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 1 \\
\hline
0 & 0 & 1 & 0 & 0 & 1 & 1
\end{array}
$$

4.6.4 Матрицы циклов и разрезов

Пусть G – (n, m)-граф, T – его остовное дерево. *Матрицей фундаментальных циклов* графа G называется булева матрица, состоящая из $\nu(G)$ строк, соответствующих фундаментальным циклам, и m столбцов, соответствующих ребрам. На пересечении i-го столбца и j-й строки имеется 1, если ребро e_i принадлежит j-му циклу, и 0 – в противном случае. Таким образом, каждая строка матрицы циклов является векторным представлением фундаментального цикла, описанным выше. При этом удобно упорядочить ребра так, чтобы в начале получаемой последовательности находились ребра, не принадлежащие остовному дереву T, и в том же порядке расположить порождаемые ими циклы. Матрица фундаментальных циклов графа, представленного на рисунке 4.48, примет тогда следующий вид:

$$\begin{array}{ccc|cccc}
e_2 & e_4 & e_7 & e_1 & e_3 & e_5 & e_6 \\
1 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 1
\end{array}$$

Левая часть такой матрицы, состоящая из $\nu(G)$ столбцов, представляет собой матрицу с единицами на главной диагонали.

Матрица фундаментальных разрезов определяется аналогично. Эта матрица имеет $n - 1$ строк и m столбцов, строки матрицы соответствуют фундаментальным разрезам, а столбцы – ребрам. Элемент на пересечении i-го столбца и j-й строки равен 1, если ребро e_i принадлежит j-му разрезу, и 0 – в противном случае. Порядок ребер при этом совпадает с их порядком в матрице циклов, а разрезы располагаются в том же порядке, что и определяющие их ребра. Матрица фундаментальных разрезов графа на рисунке 4.48 имеет следующий вид:

$$\begin{array}{ccc|cccc}
e_2 & e_4 & e_7 & e_1 & e_3 & e_5 & e_6 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 1
\end{array}$$

Как видно из данного примера, правая часть матрицы фундаментальных разрезов, состоящая из $\rho(G)$ столбцов, является единичной матрицей, а левая часть совпадает с транспонированной правой частью матрицы фундаментальных циклов. Таким образом, нахождение базиса циклов и нахождение базиса разрезов являются двойственными по отношению друг к другу задачами. Любая из представляющих эти базисы матриц получается из другой путем транспонирования.

Интересно заметить, что если приписать строки матрицы фундаментальных разрезов к матрице фундаментальных циклов, то получим квадратную матрицу, симметричную относительно главной диагонали, состоящей из единиц. Эту матрицу можно разделить на такие четыре части, что левая верхняя и правая нижняя части представляют собой единичные матрицы (с единицами на главной диагонали и только на ней), а каждая из остальных двух частей является транспонированным вариантом другой:

e_2	e_4	e_7	e_1	e_3	e_5	e_6
1	0	0 |	1	1	0	1
0	1	0 |	0	1	0	1
0	0	1 |	0	0	1	1
1	0	1 |	1	0	0	0
1	1	0 |	0	1	0	0
0	0	0 |	0	0	1	0
1	1	1 |	0	0	0	1

4.7 Числа графов

Приведем некоторые инварианты (n, m)-графа $G = (V, E)$ с k компонентами связности.

1. Степень однородного графа – степень вершин однородного графа. Например, полный граф – это однородный граф степени $n – 1$.

2. Цикломатическое число графа $\nu(G) = m - n + k$. Число ребер, которое необходимо удалить из графа, чтобы он не имел циклов.

3. Хроматическое число графа $\chi(G)$. Минимальное число цветов, в которые можно раскрасить вершины графа так, чтобы любые две смежные вершины имели разные цвета.

4. Кликовое число или плотность графа $\varphi(G)$ – число вершин наибольшей клики графа.

5. Число независимости графа $\beta_0(G)$ (или число внутренней устойчивости) – число вершин наибольшего независимого множества.

6. Число паросочетания графа $\beta_1(G)$ – число ребер в наибольшем паросочетании графа.

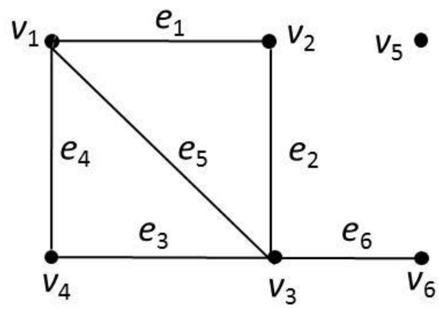
7. Число доминирования (или число внешней устойчивости) – мощность наименьшего доминирующего множества.

8. Число вершинного покрытия графа $\alpha_0(G)$ – число вершин в наименьшем покрытии графа.

9. Число реберного покрытия графа $\alpha_1(G)$ – число ребер в наименьшем покрытии графа.

Задания

1. Для приведенного ниже графа выполнить следующее:



– задать этот граф матрицами смежности и инцидентности, списком ребер;

– найти окрестности и степени вершин v_1, v_3, v_6, v_5;

– привести некоторый подграф графа; подграф, порожденный некоторым множеством вершин; некоторый остовный подграф;

– определить, является ли граф связным и сколько компонент связности он имеет;

– определить, имеет ли этот граф мост и точку сочленения;

– определить, является ли граф однородным;

– найти некоторый маршрут из вершины v_1 в вершину v_6, цепь, простую цепь; пояснить разницу между ними;

– найти расстояние между вершинами v_1 и v_6.

2. Орграф $G = (V, A)$ задан матрицей смежности:

v_1	v_2	v_3	v_4	v_5	
0	1	0	0	1	v_1
0	0	1	1	0	v_2
0	1	0	1	0	v_3
0	0	0	0	1	v_4
0	1	0	0	0	v_5

– задать граф $G = (V, A)$ в графическом виде и матрицей инцидентности;

– найти полуокрестности и полустепени исхода и захода вершин v_2 и v_3 графа, а также их степени;

– определить, является ли граф слабо связным, сильно связным;

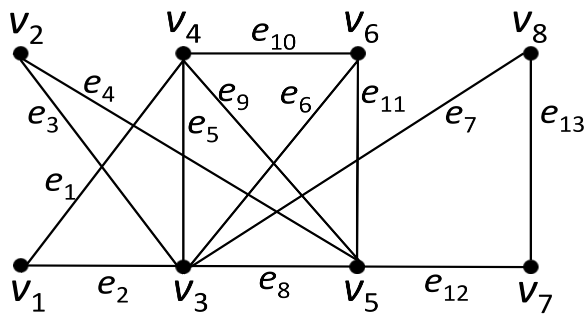
– найти расстояние между вершинами v_1 и v_5;

– найти основание $H = (V, E)$ графа $G = (V, A)$, задать его в графическом виде и матрицей смежности;

– найти дополнение графа $H = (V, E)$ и задать его в графическом виде;

– найти подграфы графа $H = (V, E)$: двудольный и однородный.

3. Для графа



– найти некоторый маршрут из вершины v_1 в вершину v_8, цепь, простую цепь, пояснить разницу между ними;

– найти расстояние между вершинами v_1 и v_8;

– найти некоторый цикл и простой цикл, пояснить разницу между ними;

– определить, является ли граф связным, транзитивным, сколько компонент связности он имеет;

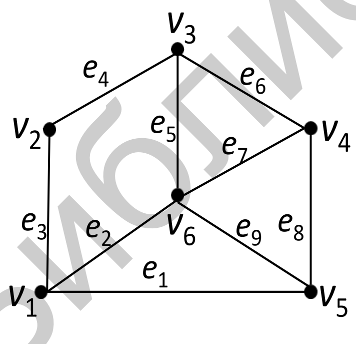
– определить кликовое число графа и предъявить соответствующую клику;

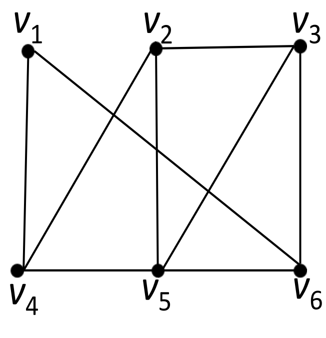
– определить, является ли граф планарным.

4. Привести пример мультиграфа, псевдографа, гиперграфа и дать их определения.

5. Построить классы изоморфных простых графов с числом вершин n = 3, 4, 5.

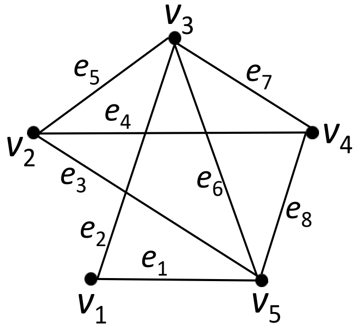
6. Установить, изоморфны ли следующие графы:

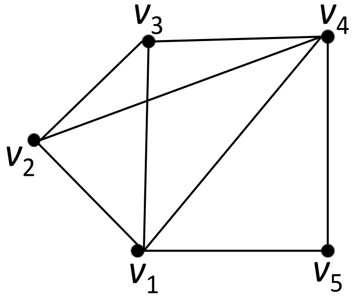


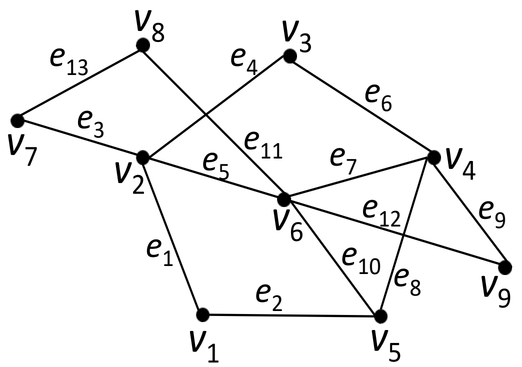


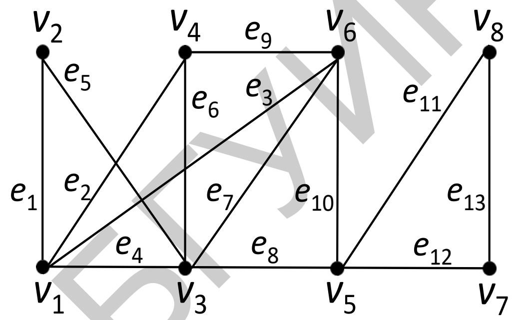
7. Построить матрицы циклов и разрезов для графов из задания 6.

8. Для каждого из следующих графов:









– установить, является ли граф эйлеровым (полуэйлеровым), в случае положительного ответа предъявить соответствующий цикл или цепь;

– установить, является ли граф гамильтоновым (полугамильтоновым), в случае положительного ответа предъявить соответствующий цикл или цепь;

– определить число доминирования графа и предъявить наименьшее доминирующее множество;

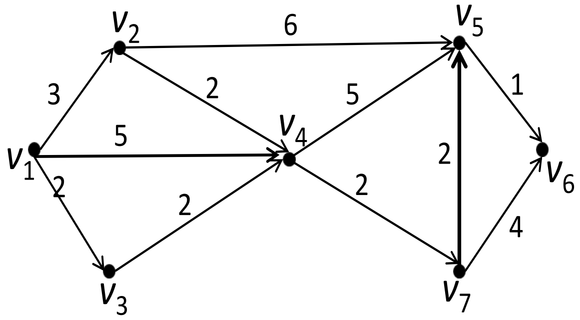
– определить число независимости графа и предъявить наибольшее независимое множество.

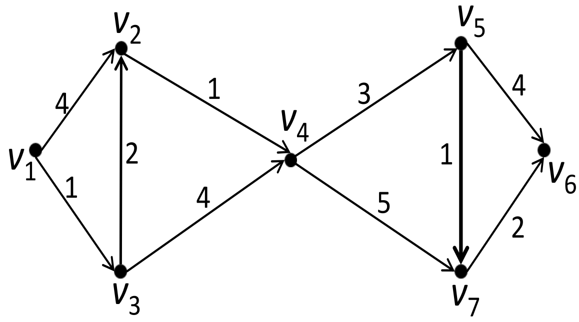
– найти вершинное покрытие графа;

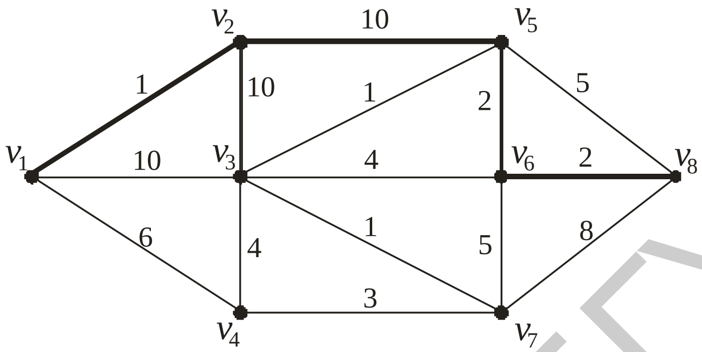
– найти минимальную раскраску графа;

– установить планарность графа.

9. Для каждого из следующих графов найти кратчайший путь между вершинами v_1 и v_6 (для последнего графа между вершинами v_1 и v_8):







10. Подсчитать число графов с одинаковым числом вершин n.

5 МАТЕМАТИЧЕСКАЯ ЛОГИКА

Основоположником логики как науки является древнегреческий философ и ученый Аристотель (384–322 гг. до н. э.), который впервые разработал теорию логического вывода одних утверждений из других. Применение в логике математических методов связывается с именем немецкого математика Г. Лейбница (1646–1716 гг.), который предложил заменить логические рассуждения вычислениями, подобно тому, как это делается в математике. Математическая логика как наука начинается с работ английского математика Джорджа Буля (1815 – 1864 гг.), труды которого положили начало алгебры логики, основными элементами которой являются двоичные переменные, принимающие значение 1 и 0 или «истина» и «ложь», и операции над ними.

5.1 Основные понятия

5.1.1 Переменные, операции

Элементами алгебры логики являются логические или булевы константы и переменные, а также операции. Имеется всего две булевы константы, которые обозначаются 0 и 1. Булевы переменные принимают значения из множества {0, 1}.

Рассмотрим шесть основных операций, определенных над булевыми константами и переменными:

– одноместная операция – отрицание (инверсия), обозначаемая как $\neg$;

– пять двухместных операций – конъюнкция, дизъюнкция, дизъюнкция с исключением («исключающее или» или сложение по модулю два), эквиваленция и импликация, обозначаемых через $\wedge$, $\vee$, $\oplus$, $\sim$ и $\to$.

1. Отрицание переменной a обозначается как $\neg a$ (или $\bar{a}$) и равно 1 тогда и только тогда, когда переменная a имеет значение 0. Отрицание – одноместная операция, в отличие от остальных, которые являются двухместными.

2. Конъюнкция переменных a и b обозначается $a \wedge b$ (или a & b) и равна 1 тогда и только тогда, когда обе переменные имеют значение 1. Символ конъюнкции иногда опускается и используется обозначение $a\,b$.

3. Дизъюнкция переменных a и b обозначается $a \vee b$ (или иногда $a + b$) и равна 1, если хотя бы одна из переменных имеет значение 1.

4. Дизъюнкция с исключением переменных a и b обозначается $a \oplus b$ и равна 1, если только одна из переменных имеет значение 1, и равна 0, если обе переменные имеют одно и то же значение.

5. Эквиваленция переменных a и b обозначается $a \sim b$ и равна 1 тогда и только тогда, когда обе переменные имеют одно и то же значение.

6. Импликация переменных a и b обозначается $a \to b$ и равна 0 только в том случае, когда a имеет значение 1, а b – значение 0, во всех остальных она равна 1.

Операции $\neg$, $\wedge$, $\vee$, $\oplus$, $\sim$ и $\to$, определенные над булевыми константами и переменными, составляют *алгебру логики*.

Результаты действия этих операций как функции логических переменных a и b представлены в таблице 5.1. Любая часть этой таблицы, задающая отдельную логическую операцию, называется *таблицей истинности*. В левой части таблицы истинности перечислены всевозможные комбинации значений логических переменных a и b – наборы значений этих переменных. Логические векторы, задающие эти наборы, лексикографически упорядочены. В правой части таблицы истинности приведены результаты выполнения соответствующих логических операций.

Таблица 5.1 – Результаты выполнения логических операций

$a\ b$	$\neg a$	$a \wedge b$	$a \vee b$	$a \oplus b$	$a \sim b$	$a \to b$
0 0	1	0	0	0	1	1
0 1	1	0	1	1	0	1
1 0	0	0	1	1	0	0
1 1	0	1	1	0	1	1

5.1.2 Формулы и функции

Пусть x_1, x_2, ..., x_n – некоторые булевы переменные, т. е. переменные, принимающие значение из двухэлементного множества $E = \{0, 1\}$. Упорядоченную совокупность булевых переменных $(x_1, x_2, \dots, x_n)$ можно рассматривать как n-компонентный *булев вектор* $\boldsymbol{x}$. Число компонент вектора определяет его *длину*, или *размерность*. При фиксации значений всех переменных получается *набор значений* переменных, задаваемый булевым вектором длиной n, состоящим из констант 0 и 1. Совокупность различных n-компонентных булевых векторов $\boldsymbol{x}$ образует множество E^n (n-ю степень множества E), называемое *булевым* *пространством* размерностью n. Нетрудно подсчитать, что мощность $|E^n|$ булева пространства E^n равна 2^n.

Функция n аргументов, обозначаемая как $y = f(x_1, x_2, ..., x_n)$, называется *булевой*, если ее аргументы и сама функция являются булевыми переменными. Булева функция задает отображение n-мерного булева пространства E^n в двухэлементное множество E: $f\colon E^n \to E$. *Областью определения* булевой функции $f(x_1, x_2, ..., x_n)$ является булево пространство E^n, а *областью значений* – E. Задание функции f на E^n разделяет его на две области: M_f^1 и M_f^0, где она принимает значения 1 и 0 соответственно. Множество M_f^1 наборов значений переменных $x_1, x_2, ..., x_n$ называется множеством *единичных* значений или *характеристическим множеством* функции $f(x_1, x_2, ..., x_n)$. Соответственно M_f^0 называется множеством *нулевых* значений функции.

Простейшим способом задания булевой функции является определенная выше таблица истинности (см. таблицу 5.1), в левой части которой в лексикографическом порядке перечислены все 2^n наборов значений переменных $x_1, x_2,\dots, x_n$, а в правой части указаны значения функции на этих наборах. Наборы, на которых значение функции f равно 1, называются *единичными* наборами функции, а наборы, на которых f равно 0, – *нулевыми*.

Используя приведенные выше операции алгебры логики можно создавать композиции логических операций и булевых переменных, выражаемые *формулами*. Например, формула

$$(a \oplus b) \sim \bar{a}$$

является композицией трех операций: дизъюнкции с исключением, эквиваленции и отрицания переменной.

Формула определяется индуктивно следующим образом:

1) символы a, b, c, … булевых переменных или констант являются формулами;

2) если A и B – формулы, то формулами являются $\bar{A}$ и $(A * B)$, где «*» – операция из множества $\{\wedge, \vee, \oplus, \sim, \to\}$;

3) других формул нет.

Формулы, состоящие из единственной переменной или константы, называются *простыми*. Индуктивное определение формулы расширяет набор простых формул за счет *составных* (сложных) формул, которые образуются путем связывания простых формул разными операциями.

Приведенное определение формулы задает правила ее образования и позволяет конструировать любые сложные формулы путем последовательного связывания их частей операциями алгебры логики. Однако получаемое таким образом представление формулы теряет наглядность из-за многочисленных скобок. Если придерживаться принятых в алгебре логики правил приоритета операций, некоторые скобки можно опускать, сохраняя при этом порядок выполнения операций. Принято разбивать операции алгебры логики на следующие четыре класса в соответствии с приоритетностью их выполнения, которая отражает относительную силу связи частей формулы операциями:

1) отрицание «$\neg$»;

2) конъюнкция «$\wedge$»;

3) дизъюнкция «$\vee$» и дизъюнкция с исключением «$\oplus$»;

4) импликация «$\to$» и эквиваленция «~».

После удаления скобок в формуле порядок выполнения операций определяется их приоритетностью: первой выполняется операция «$\neg$», обладающая первым приоритетом, затем – «$\wedge$», далее – операции «$\vee$» и «$\oplus$», имеющие третий приоритет и, наконец, операции «~» и «$\to$», имеющие последний, четвертый приоритет.

Пользуясь этими правилами, можно оставлять скобки только там, где это может изменить установленный порядок выполнения. Например, скобки можно опустить в формулах $(a \wedge (\bar{b})) \to (c \vee d)$ и $(a \vee b) \vee c$, получив $a \wedge \bar{b} \to c \vee d$ и $a \vee b \vee c$. Но нельзя опустить скобки в формулах $\neg(a \vee b)$, $a \wedge (b \to c)$ и $a \oplus (b \vee c)$.

Порядок выполнения операций имеет существенное значение при вычислении значений формулы. Например, для формулы $\bar{a}\, b \to \bar{c}\, d \vee c\, \bar{d}$ при некоторых заданных значениях a, b, c и d сначала определяются значения $\bar{a}$, $\bar{c}$, $\bar{d}$, затем вычисляются $\bar{a}\, b$, $\bar{c}\, d$, $c\, \bar{d}$, после этого $\bar{c}\, d \vee c\, \bar{d}$ и только потом находится результат действия импликации, что равносильно следующей расстановке скобок: $\bar{a}\, b \to \bar{c}\, d \vee c\, \bar{d} = (((\bar{a})\, b) \to (((\bar{c})\, d) \vee (c(\bar{d}))))$.

Всякую формулу алгебры логики можно рассматривать как представление некоторой функции $f(x_1, x_2, \ldots, x_n)$, аргументами x_i которой являются булевы переменные данной формулы. Значение функции определяется значениями входящих в нее переменных и типами логических операций. Например, значения функции $f(a, b)$, представляемой формулой $a \oplus b$, на всех наборах значений ее аргументов приведены в пятом столбце таблицы 5.1.

Составную формулу можно рассматривать как суперпозицию элементарных функций, соответствующих операциям, входящим в формулу. Под суперпозицией функций понимается использование символов одних функций в качестве аргументов некоторых других функций. Другими словами, функция $f(x_1, x_2, \ldots, x_n)$ называется *суперпозицией* функций $f_0(x_1, x_2, \ldots, x_n)$, $f_1(x_1, x_2, \ldots, x_n)$, $f_2(x_1, x_2, \ldots, x_n)$, …, $f_m(x_1, x_2, \ldots, x_n)$, если

$$f(x_1, x_2, \ldots, x_n) = f_0(f(x_1, x_2, \ldots, x_n), f_2(x_1, x_2, \ldots, x_n), \ldots, f_m(x_1, x_2, \ldots, x_n)).$$

Например, рассмотрим формулу, задающую функцию $f(x, y, z)$:

$$f(x, y, z) = (x \oplus y)\, z \vee x\, y.$$

Эту формулу можно представить в виде суперпозиции $f(x, y, z) = g(g_1, g_2)$ функций g, g_1, g_2, g_3, где

$$g = g(g_1, g_2) = g_1 \vee g_2;$$

$$g_1 = g_1(g_3, z) = g_3 \wedge z;$$

$$g_2 = g_2(x, y) = x \wedge y;$$

$$g_3 = g_3(x, y) = x \oplus y.$$

5.1.3 Вычисление значения формулы

Вычисление значений функции по формуле проиллюстрируем на примере следующей формулы:

$$F = a\, b \oplus c \vee \bar{b} \sim (b \to a \vee \bar{c}). \tag{5.1}$$

Перед вычислением значения функции расставим скобки в формуле в соответствии с приоритетом операций:

$$F = ((((a\, b) \oplus c) \vee (\bar{b})) \sim (b \to (a \vee (\bar{c})))).$$

Рассмотрим три следующих способа вычисления значения формулы:

1) по табличному представлению;

2) по представлению в виде дерева;

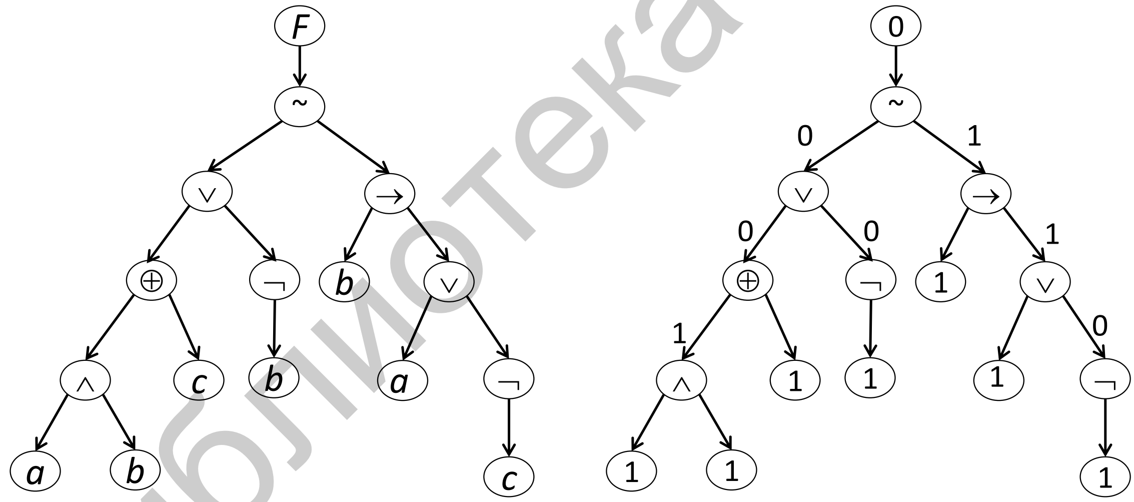
3) по польской записи.

Вычисление по табличному представлению. Промежуточные результаты вычисления значения функции и окончательный результат представлены в таблице 5.2.

Вычисление по представлению в виде дерева. Формулу можно представить в виде ориентированного дерева с корнем (рисунок 5.1). Корню дерева поставим в соответствие значение формулы F. С неконцевыми вершинами связаны операции, входящие в формулу. Концевым вершинам (листьям дерева, т. е. тем вершинам, из которых не исходит ни одна дуга) соответствуют переменные, входящие в формулу. Из корня исходит единственная дуга, соединяющая его с вершиной, которой соответствует операция, выполняемая в последнюю очередь. Из каждой неконцевой вершины дерева исходят дуги, связывающие ее с вершинами, которым приписаны либо операции, выполнение которых предшествует данной, либо переменные, над которыми выполняется данная операция.

Таблица 5.2 – Вычисление значения функции $F = a\,b \oplus \bar{c}\,e \sim (b \to d \vee \bar{e})$

a b c	$a \wedge b$	$(a\,b) \oplus c$	$\bar{b}$	$(a\,b \oplus c) \vee (\bar{b})$	$\bar{c}$	$a \vee (\bar{c})$	$b \to (a \vee \bar{c})$	F
0 0 0	0	0	1	1	1	1	1	1
0 0 1	0	1	1	1	0	0	1	1
0 1 0	0	0	0	0	1	1	1	0
0 1 1	0	1	0	1	0	0	0	0
1 0 0	0	0	1	1	1	1	1	1
1 0 1	0	1	1	1	0	1	1	1
1 1 0	1	1	0	1	1	1	1	1
1 1 1	1	0	0	0	0	1	1	0



а *б*

а – графическое представление формулы $F = \neg((a \oplus b)\,c) \to \neg\, a \vee b\,c$;
б – процесс вычисления значения формулы F

Рисунок 5.1 – Вычисление значения формулы по представлению в виде дерева

С помощью построенного дерева можно вычислить значение соответствующей формулы при заданных значениях ее переменных. Процесс вычисления значения формулы производится снизу вверх по дереву с приписыванием вычисляемых значений дугам, предшествуемым выполняемым операциям. На рисунке 5.1, *б* отображен процесс вычисления значения формулы (5.1) при следующих значениях входящих в нее переменных: $a = 1$, $b = 1$ и $c = 1$.

Вычисление по польской записи. Для вычисления значения формул с помощью вычислительной машины удобна *польская запись*, которая является бесскобочным видом формулы. В ней используется префиксная нотация операций, т. е. символ операции располагается слева от операндов: $*AB$ или $\neg A$, где $* \in \{\wedge, \vee, \oplus, \sim, \rightarrow\}$, A и B – любые формулы того же вида. В процессе вычисления значения формулы по польской записи символы просматриваются справа налево, порядок размещения операций обратный по отношению к их выполнению и выполняемая последней операция должна располагаться слева.

Например, польская запись приведенной выше формулы (5.1)

$$F = ((((a\,b) \oplus c) \vee (\ \bar{b})) \sim (b \rightarrow (a \vee (\ \bar{c}))))$$

будет иметь следующий вид:

$$\sim \vee \oplus \wedge\, a\, b\, c \neg b \rightarrow b \vee a \neg c. \tag{5.2}$$

В процессе вычисления символы операций и значения переменных помещаются в *стек*. Записи помещаются в стек и удаляются из него с одного конца, называемого *вершиной стека*, которая располагается справа. Работа со стеком организована по принципу «последним вошел, первым вышел» (LIFO – Last In First Out). Символы считываются слева до тех пор, пока не встретится символ операции. Ее операндами становятся ближайшие к ней справа константы (одна или две в зависимости от местности операции). После выполнения операции ее символ и операнды заменяются в стеке результатом выполнения этой операции. Просмотр стека и выполнение операций продолжается, пока ни будет достигнут конец и получен единственный элемент стека, который и представляет результат вычисления значения формулы.

Ниже приведена последовательность изменений содержимого стека при вычислении значения формулы, заданной польской записью (5.2), при следующих значениях переменных: $a = 1$, $b = 1$ и $c = 1$.

$\sim \vee \oplus \wedge 1\ 1\ 1 \neg 1 \to 1 \vee 1\ \underline{\neg 1}$;

$\sim \vee \oplus \wedge 1\ 1\ 1 \neg 1 \to 1\ \underline{\vee 1\ 0}$;

$\sim \vee \oplus \wedge 1\ 1\ 1 \neg 1\ \underline{\to 1\ 1}$;

$\sim \vee \oplus \wedge 1\ 1\ 1\ \underline{\neg 1}\ 1$;

$\sim \vee \oplus\ \underline{\wedge 1\ 1}\ 1\ 0\ 1$;

$\sim \vee\ \underline{\oplus 1\ 1}\ 0\ 1$;

$\sim \underline{\vee 0\ 0}\ 1$;

$\underline{\sim 0\ 1}$;

0.

После окончания просмотра формулы единственный элемент стека представляет результат вычисления. Таким образом, $F = 0$ при $a = 1$, $b = 1$ и $c = 1$.

Польскую запись формулы легко получить, совершая обход представляющего ее дерева из корня сверху вниз и слева направо. При этом последовательно записываются символы проходимых вершин (если они встречаются впервые).

Для того чтобы получить польскую запись, исходя непосредственно из формулы, можно сначала расставить и пронумеровать скобки, устанавливающие порядок выполнения операций в формулах. Затем совершить просмотр скобок, начиная с внешних, слева направо. Например, рассматриваемая выше формула после расстановки скобок примет следующий вид:

$$F = ({}_1({}_2({}_3({}_4 a\, b) \oplus c) \vee ({}_5\ \bar{b})) \sim ({}_6 b \to ({}_7 a \vee ({}_8\ \bar{c})))) .$$

После просмотра скобок в порядке их нумерации и записи операций и переменных получаем искомую запись:

$$\sim \vee \oplus \wedge a\, b\, c \neg b \to b \vee a \neg c.$$

5.2 Отношения между формулами

Важнейшими отношениями между формулами являются отношения равносильности и формальной импликации.

5.2.1 Равносильность

Формулы A и B *равносильны* или логически эквивалентны, если они представляют одну и ту же функцию, или, другими словами, формулы A и B принимают одинаковые значения при любых значениях входящих в них переменных. Равносильность формул A и B обозначается как $A = B$ или $A \Leftrightarrow B$.

Например,

$$(a \oplus b) = a\,\bar{b} \vee \bar{a}\,b;$$

$$(w \to v)\,(v \to w) \;=\; w \sim v.$$

Если обозначить через M_A и M_B множество наборов значений, на которых функции, представляемые формулами A и B, принимают значение 1, то $M_A = M_B$, если A и B равносильны. Соответственно, доказать равносильность формул можно, например, путем вычисления значений каждой из формул по их табличным представлениям и сравнения результатов, как это показано в таблице 5.3.

Таблица 5.3 – Доказательство равносильности формул $F_1 = (w \to v)(v \to w)$ и $F_2 = w \sim v$

v	w	$w \to v$	$v \to w$	F_1	F_2
0	0	1	1	1	1
0	1	0	1	0	0
1	0	1	0	0	0
1	1	1	1	1	1

Нетрудно заметить, что отношение равносильности формул обладает следующими свойствами:

– симметричности: если $A = B$, то и $B = A$;

– транзитивности: если $A = B$ и $B = C$, то и $A = C$;

– рефлексивности: $A = A$.

Таким образом, отношение равносильности формул является отношением эквивалентности. Это обеспечивает взаимозаменяемость равносильных формул. Можно заменить некоторую формулу или ее часть на равносильную, при этом формула будет задавать ту же самую функцию. Это свойство позволяет упрощать формулу путем замены ее на эквивалентную, имеющую более простое представление.

При определении отношений между формулами не обязательно предполагать, что они определены над одними и теми же переменными. Если некоторая переменная входит только в одну из формул, то отношение имеет место при любых значениях этой переменной. В этом случае значение соответствующей формулы, не содержащей некоторую переменную, не зависит от нее. Например, рассмотрим пару равносильных формул (таблица 5.4):

$$(a\,\bar{b} \vee c) \wedge b = bc.$$

Первая из формул зависит от переменных a, b и c, а вторая – только от b и c.

Таблица 5.4 – Доказательство равносильности формул
$F_1 = (a\,\bar{b} \vee c)b$ и $F_2 = bc$

a	b	c	$\bar{b}$	$a\,\bar{b}$	$a\,\bar{b} \vee c$	F_1	$F_2 = bc$
0	0	0	1	0	0	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	0	0	0
0	1	1	0	0	1	1	1
1	0	0	1	1	1	0	0
1	0	1	1	1	1	0	0
1	1	0	0	0	0	0	0
1	1	1	0	0	1	1	1

5.2.2 Формальная импликация

Формулы A и B находятся в отношении формальной импликации, точнее, *A имплицирует B*, если формула B принимает значение 1 на всех наборах значений переменных, на которых значение 1 принимает формула A. В таких случаях говорят еще, что формула *B логически следует* из формулы A. Факт наличия такой связи между формулами A и B обозначается как $A \Rightarrow B$.

Например,

$$a \wedge b \;\Rightarrow\; a \vee b;$$

$$a \sim b \;\Rightarrow a \to b.$$

Если обозначить через M_A и M_B множество наборов значений, на которых функции, представляемые формулами A и B, принимают значение 1, то справедливо $M_A \subseteq M_B$, если $A \Rightarrow B$ (рисунок 5.2). В этом смысле A является *импликантой B*, а B, в свою очередь, – *имплицентой A*.

Доказать, что формула A имплицирует B, можно, например, путем вычисления значений каждой из формул по их табличным представлениям и сравнения результатов, как это показано в таблице 5.5.

Нетрудно заметить следующую связь между отношениями равносильности и формальной импликации: если формулы A и B следуют друг из друга, т. е. $A \Rightarrow B$ и $B \Rightarrow A$, то они равносильны: $A \Leftrightarrow B$.

Например, если $(a \oplus b) \Rightarrow \neg(a \sim b)$ и $\neg(a \sim b) \Rightarrow (a \oplus b)$, то очевидно, что между этими формулами имеет место отношение равносильности:

$$(a \oplus b) \Leftrightarrow \neg(a \sim b).$$

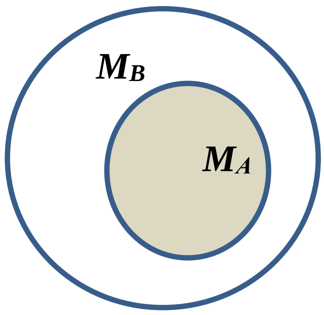


Рисунок 5.2 – Теоретико-множественная интерпретация отношения $A \Rightarrow B$

Таблица 5.5 – Доказательство
$a \sim b \;\Rightarrow\; a \to b$

a	b	$a \sim b$	$a \to b$
0	0	1	1
0	1	0	1
1	0	0	0
1	1	1	1

5.2.3 Выполнимость и общезначимость формул

Формула, которая принимает значение 1 хотя бы на одном наборе значений ее переменных, называется *выполнимой*. Выполнимая формула, которая не принимает значение 0 ни при каких значениях переменных, называется *тавтологией*.

Например, примерами тавтологий служат следующие формулы, принимающие значение 1 при всех значениях своих переменных:

$$A = (a \wedge b) \to (a \vee b);$$

$$B = a \vee b \vee \bar{a}\ \bar{b},$$

а следующие выполнимые формулы тавтологиями не являются:

$$C = a \sim b;$$

$$D = b \wedge c.$$

Формула, которая принимает значение 0 на всех наборах значений переменных, называется *невыполнимой* или *противоречием*. В качестве примеров таких формул можно привести

$$E = (b \vee \bar{b}) \rightarrow (a \wedge \bar{a});$$
$$F = (a \sim b) \wedge (a \oplus b);$$
$$G = a \wedge \bar{a}.$$

Проверить формулу на выполнимость или доказать, что она является тавтологией, противоречием, можно, например, с помощью вычисления ее значений по табличному представлению, как это показано в таблице 5.6.

Таблица 5.6 – Проверка формул $A = (a \wedge \bar{a}) \rightarrow (b \vee \bar{b})$ и $E = (b \vee \bar{b}) \rightarrow (a \wedge \bar{a})$ на выполнимость

a	b	$\bar{a}$	$\bar{b}$	$a\ \bar{a}$	$b \vee \bar{b}$	A	E
0	0	1	1	0	1	1	0
0	1	1	0	0	1	1	0
1	0	0	1	0	1	1	0
1	1	0	0	0	1	1	0

Между отношением равносильности формул и операцией эквиваленции существует следующая связь: если формулы A и B равносильны, то $A \sim B$ является тавтологией. И наоборот: если формула $A \sim B$ – тавтология, то формулы A и B равносильны. Справедливость этого утверждения непосредственно вытекает из определения операции эквиваленции.

Аналогичная связь имеется и между отношением формальной импликации и операцией импликации. Если $A \Rightarrow B$, то $A \rightarrow B$ является тавтологией. Например, из того факта, что $a \wedge b \Rightarrow a \vee b$, следует, что формула $a \wedge b \rightarrow a \vee b$ является тавтологией (таблица 5.7).

Таблица 5.7 – Проверка формул $a \wedge b \Rightarrow a \vee b$
и $a \wedge b \rightarrow a \vee b$ на тавтологию

a	b	$a\,b$	$a \vee b$	$a \wedge b \rightarrow a \vee b$
0	0	0	0	1
0	1	0	1	1
1	0	0	1	1
1	1	1	1	1

5.3 Булева алгебра

5.3.1 Основные законы булевой алгебры

Алгебра логики на основе трех операций – отрицания, конъюнкции и дизъюнкции – называется *булевой алгеброй*, если для нее выполняются следующие законы (равносильности, аксиомы булевой алгебры):

1. *Идемпотентность*: $x \vee x = x$; $\quad x\,x = x$.
2. *Коммутативность*: $x \vee y = y \vee x$; $\quad x\,y = y\,x$.
3. *Ассоциативность*: $x \vee (y \vee z) = (x \vee y) \vee z$; $\quad x\,(y\,z) = (x\,y)\,z$.
4. *Дистрибутивность*: $x\,(y \vee z) = x\,y \vee x\,z$; $\quad x \vee y\,z = (x \vee y)\,(x \vee z)$.
5. *Законы де Моргана*: $\overline{x \vee y} = \bar{x}\;\bar{y}$; $\quad \overline{x\,y} = \bar{x} \vee \bar{y}$.
6. *Двойное отрицание (инволюция)*: $\bar{\bar{x}} = x$.
7. *Законы операций с константами*: $x \wedge 1 = x$; $\quad x \vee 1 = 1$;
$x \wedge 0 = 0$; $\quad x \vee 0 = x$;
$x \wedge \bar{x} = 0$; $\quad x \vee \bar{x} = 1$.

Операции отрицания, дизъюнкции и конъюнкции образуют *булев базис*, а формулы, включающие только эти операции, называются *булевыми*. Любая булева функция может быть представлена булевой формулой.

Для булевой алгебры справедлив *принцип двойственности*: любая формула из пары равносильных (в том числе представляющих некоторый закон этой алгебры) получается из другой путем следующих замен:

– символов конъюнкции на символы дизъюнкции и наоборот;

– констант 1 на константы 0 и наоборот.

Например, если имеет место равносильность $x \vee x\,y = x$, то справедливо и $x\,(x \vee y) = x$.

Кроме приведенных выше законов булевой алгебры, выделяют также следующие равносильные формулы (законы):

1. *Закон поглощения*: $x \vee x\,y = x$; $\quad x\,(x \vee y) = x$.

2. *Закон простого склеивания*: $x\,y \vee x\,\bar{y} = x$; $\quad (x \vee y)\,(x \vee \bar{y}) = x$.

3. *Закон обобщенного склеивания*: $x\,y \vee \bar{x}\,z = x\,y \vee \bar{x}\,z \vee y\,z$;

$(x \vee y)\,(\bar{x} \vee z) = (x \vee y)\,(\bar{x} \vee z)\,(y \vee z)$.

4. *Закон упрощения*: $x \vee \bar{x}\,y = x \vee y$; $\quad x\,(\bar{x} \vee y) = x\,y$.

5. *Контрапозиция*: $x \to y = \bar{y} \to \bar{x}$.

6. *Экспортация*: $x\,y \to z = x \to (y \to z)$.

7. Выражение операций алгебры логики через булевы операции:

$$x \to y = \bar{x} \vee y;$$

$$x \sim y = x\,y \vee \bar{x}\,\bar{y} = (x \vee \bar{y}) \wedge (\bar{x} \vee y);$$

$$x \oplus y = x\,\bar{y} \vee \bar{x}\,y = (x \vee y)\,(\bar{x} \vee \bar{y}).$$

8. Выражение операций алгебры логики через другие операции:

$$x \to y = \bar{x}\,y \oplus x \oplus 1;$$

$$x \sim y = (x \to y) \wedge (y \to x) = \overline{x \oplus y} = 1 \oplus x \oplus y;$$

$$x \oplus y = \overline{x \sim y};$$

$$\bar{x} = x \oplus 1;$$

$$x\,y = \overline{x \to \bar{y}};$$

$$x \vee y = \bar{x} \to y.$$

В справедливости приведенных равносильностей легко убедиться, построив и сравнив таблицы истинности.

5.3.2 Интерпретации булевой алгебры

Абстрактная булева алгебра имеет ряд интерпретаций, используемых в различных приложениях.

1. *Булева алгебра множеств*. Переменными в этой алгебре являются подмножества некоторого универсального множества U, константами, аналогичными константам 1 и 0, служат множества U и $\varnothing$. Все законы, приведенные ранее для алгебры множеств (см. пункт 1.3.2), совпадают с основными законами абстрактной булевой алгебры, если операцию дополнения множества заменить на операцию отрицания, а операции пересечения $\cap$ и объединения $\cup$ множеств – соответственно на операции конъюнкции $\wedge$ и дизъюнкции $\vee$.

2. *Булева алгебра высказываний* является одной из интерпретаций абстрактной булевой алгебры. Здесь переменными являются высказывания, принимающие истинные или ложные значения, которые соответствуют константам 1 и 0. Символы операций и их названия в данном случае совпадают с операциями булевой алгебры (см. пункт 5.3.1).

3. *Алгебра переключательных схем*. Переменным этой алгебры соответствуют элементы переключательной схемы – переключатели. Переключательный элемент, состояние которого представляется булевой переменной a, может быть замкнут, тогда через него течет ток и $a = 1$. Если он разомкнут, то тока нет и $a = 0$. По состояниям переключателей в схеме можно определить, проходит ли по данной схеме ток. На рисунке 5.3, *а* изображено последовательное соединение двух переключателей a и b. Данная схема будет пропускать ток в том и только в том случае, когда оба переключателя замкнуты, т. е. если $a \wedge b = 1$. На рисунке 5.3, *б* изображено параллельное соединение переключателей a и b. Ток будет протекать, если замкнут хотя бы один из переключателей, т. е. если $a \vee b = 1$.

а *б*

Рисунок 5.3 – Примеры последовательного (*а*) и параллельного (*б*) соединений переключателей

Переключатели можно связать таким образом, чтобы они замыкались и размыкались одновременно. Такие переключатели обычно обозначаются одним и тем же символом. Каждому переключателю можно поставить в соответствие другой переключатель так, чтобы когда один из них замкнут, другой был разомкнут. Если один из них обозначить буквой a, то другой примет обозначение $\bar{a}$. В схеме на рисунке 5.4 пойдет ток, если $a\,b \vee b\,\bar{c} \vee \bar{a}\,b = 1$. Левая часть этого уравнения представляет структуру схемы.

5.3.3 Равносильные преобразования формул

Назовем *термом* любую часть формулы, являющуюся, в свою очередь, формулой. Например, фрагменты $a,\ b,\ a \to c,\ (a \to b) \wedge (b \to c)$ формулы

$$(a \to b) \wedge (b \to c) \to (a \to c)$$

являются термами, тогда как «$\to$», «$\to c$» ими не являются.

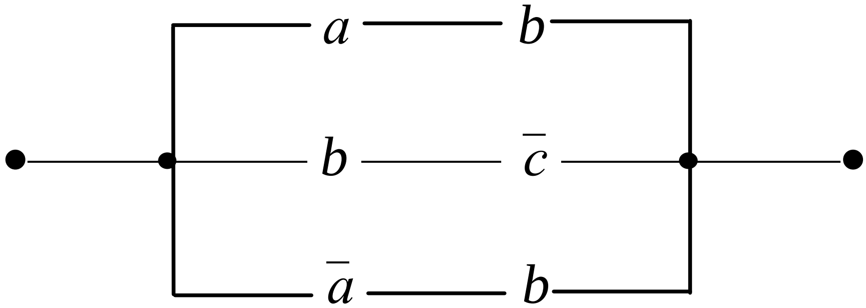


Рисунок 5.4 – Пример переключательной схемы

Выделяются следующие типы преобразований формул, сохраняющих их равносильность.

1. *Глобальная подстановка термов*. Подстановка в формулы, находящиеся в отношении равносильности, любого терма вместо всех вхождений некоторой переменной сохраняет отношение равносильности между этими формулами.

Например, подстановка вместо переменной x терма $\overline{a \vee b}$ в равносильные формулы $\overline{\bar{x} \wedge y}$ и $x \vee \bar{y}$ приводит к равносильным же формулам

$$\overline{(a \vee b) \wedge y} = \overline{a \vee b} \vee \bar{y}.$$

Существенным в данном случае является то, что равносильность формул сохраняется, если все до одного вхождения переменной заменяются тем же термом.

2. *Двухстороннее отрицание*. Равносильные формулы при их отрицании порождают также равносильные формулы – отношение равносильности между формулами сохраняется, т. е. если $A = B$, то и $\bar{A} = \bar{B}$.

Например, легко убедиться путем построения и сравнения таблиц истинности в том, что из $a\,b \wedge a\,\bar{b} = a$ вытекает $\overline{a\,b \wedge a\,\bar{b}} = \bar{a}$.

Но отрицание формул левой и правой частей формальной импликации меняет местами причину и следствие, т. е. если $A \Rightarrow B$, то $\bar{B} \Rightarrow \bar{A}$. Например, легко убедиться в том, что из $a\,b \Rightarrow a \vee b$ вытекает, что $\overline{a \vee b} \Rightarrow \overline{a\,b}$.

3. *Двухсторонняя замена на двойственные функции.* Функции, двойственные равносильным функциям, также равносильны. Это проявляется и в основных законах булевой алгебры.

Например, в законе дистрибутивности $x(y \vee z) = x y \vee x z$ при замене формул на двойственные равносильность сохраняется: $x \vee y z = (x \vee y)(x \vee z)$.

4. *Замена подформулы на равносильную*. При замене в формуле любой ее подформулы на равносильную получается формула, равносильная исходной. Например, при замене в формуле

$$A = \overline{((a \vee b) \wedge y)}$$

подформулы $a \vee b$ на равносильную ей $\overline{\bar{a} \wedge \bar{b}}$ получается формула, равносильная исходной:

$$\overline{((a \vee b) \wedge y)} = \overline{\overline{\bar{a} \wedge \bar{b}} \wedge y} = (\bar{a} \wedge \bar{b}) \vee \bar{y} = \overline{a \vee b} \vee \bar{y} = A.$$

Приведенные выше равносильности 1 – 4 алгебры логики можно вывести путем тождественных (равносильных) преобразований одной из их частей путем замены в них подформул, составляющих законы булевой алгебры.

Действительно, *закон поглощения* $x \vee x y = x$ выводится следующим образом:

$$x \vee x y = x \wedge 1 \vee x \wedge y = x(1 \vee y) = x \wedge 1 = x.$$

Закон поглощения, двойственный выведенному, может быть получен из последнего заменой операций $\vee$ на $\wedge$ и наоборот (в силу справедливости принципа двойственности для булевой алгебры) или также путем тождественных преобразований:

$$x(x \vee y) = x x \vee x y = x \vee x y = x.$$

Закон *простого склеивания* выводится с использованием закона дистрибутивности конъюнкции относительно дизъюнкции и операций с константами:

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

Для вывода закона *обобщенного склеивания* проще преобразовывать не левую, а правую часть равносильности:

$$x y \vee \bar{x} z \vee y z = x y \vee \bar{x} z \vee y z(x \vee \bar{x}) = x y \vee \bar{x} z \vee x y z \vee \bar{x} y z =$$

$$= (x y \vee x y z) \vee (\bar{x} z \vee \bar{x} y z) = x y(1 \vee z) \vee \bar{x} z(1 \vee y) = x y \vee \bar{x} z.$$

Закон *удаления литерала* выводится с помощью закона дистрибутивности дизъюнкции относительно конъюнкции:

$$x \vee \bar{x} y = (x \vee \bar{x})(x \vee y) = x \vee y.$$

Пользуясь вышеприведенными формулами, построим булево выражение, эквивалентное следующей логической формуле, и упростим его:

$$((x \to y) \vee (x \oplus z))\ \bar{y} = (\bar{x} \vee y \vee x\,\bar{y} \vee \bar{x}\,y)\ \bar{y} = \bar{x}\ \bar{y} \vee x\ \bar{y} = \bar{y}.$$

В ходе равносильных преобразований формулы использовались следующие тождества, которые становятся очевидными при анализе таблиц истинности для этих операций (таблица 5.8):

$$x \oplus 1 = \bar{x};$$

$$x \to y = \bar{x} \vee y;$$

$$x \sim y = \bar{x}\ \bar{y} \vee xy\,.$$

Таблица 5.8 – Соотношения между логическими операциями

x	y	$\bar{x}$	$x \oplus 1$	$\bar{x} \vee y$	$x \to y$	xy	$\bar{x}\,\bar{y}$	$xy \vee \bar{x}\,\bar{y}$	$x \sim y$
0	0	1	1	**1**	**1**	0	1	**1**	**1**
0	1	1	1	**1**	**1**	0	0	**0**	**0**
1	0	0	0	**0**	**0**	0	0	**0**	**0**
1	1	0	0	**1**	**1**	1	0	**1**	**1**

В общем случае преобразование формул позволяет приводить их к более простому виду. Продемонстрируем применение приведенных выше законов на примере преобразования следующей формулы:

$$A = (b \to a) \wedge ((b \oplus 1) \wedge \bar{c} \sim c).$$

Ниже приведены равносильные преобразования формулы A. Используемые при этом равносильности приведены слева, а результат их применения к формуле A показан справа:

$x \to y = \bar{x} \vee y;$	$(\bar{b} \vee a) \wedge ((b \oplus 1) \wedge \bar{c} \sim c);$
$x \oplus 1 = \bar{x};$	$(\bar{b} \vee a) \wedge (\bar{b}\ \bar{c} \sim c);$
$x \sim y = \bar{x}\ \bar{y} \vee xy \vee \bar{x}\ \bar{y};$	$(\bar{b} \vee a) \wedge (\overline{\bar{b}\,\bar{c}}\ \bar{c} \vee \bar{b}\ \bar{c}\,c) =$
	$= (\bar{b} \vee a) \wedge (b \vee c)\ \bar{c} = b\ \bar{c} =$
	$= (\bar{b} \vee a) \wedge (b\ \bar{c}) = a\,b\ \bar{c}\,.$

Таким образом, имеем

$$A = (b \to a) \wedge ((b \oplus 1) \wedge \bar{c} \sim c) = ab\ \bar{c}.$$

5.4 Нормальные формы булевой алгебры

Рассмотрим специальные виды формул в булевой алгебре, так называемые *нормальные формы*, к которым можно привести любую формулу этой алгебры. В основе нормальных форм лежат многоместные операции дизъюнкции и конъюнкции, которые являются обобщениями двуместных:

$$a \vee b \vee c \vee d = \vee\,(a, b, c, d);$$
$$a \wedge b \wedge c \wedge d = \wedge\,(a, b, c, d).$$

Первая формула принимает значение 1, если хотя бы одна из ее переменных имеет значение 1, вторая – если значением 1 обладают все переменные; в противном случае формулы принимают значение 0. Другими словами, значение дизъюнкции равно максимальному из значений ее переменных, значение конъюнкции – минимальному.

5.4.1 Дизъюнктивная нормальная форма

Введем в рассмотрение простейшие функции – булевы переменные a, b, c,… и их инверсии $\bar{a}$, $\bar{b}$, $\bar{c}$, …, которые будем называть положительными и отрицательными *литералами* или просто литералами (или буквами).

Элементарной конъюнкцией называется многоместная конъюнкция попарно различных литералов. К числу элементарных отнесем также конъюнкции, состоящие из одного литерала, а также константу единица «1» – конъюнкцию пустого числа литералов. Число литералов элементарной конъюнкции назовем ее *рангом*.

Элементарными конъюнкциями являются, например, выражения

$$1,\ x,\ \ \bar{x},\ xy,\ x\bar{y}z,\ \ \bar{x}\,\bar{y}\,\bar{z},$$

имеющие ранги соответственно 0, 1, 1, 2, 3, 3. В то же время элементарными конъюнкциями не являются, например, следующие выражения:

$$0,\ \overline{x\,y}\,,\ \ xyx,\ \ \bar{x}xy.$$

Дизъюнктивной нормальной формой (ДНФ) называется формула, представляющая дизъюнкцию элементарных конъюнкций. Входящие в ДНФ элементарные конъюнкции называются также *конъюнктивными термами*.

По определению следующие выражения представляют собой ДНФ:

$$xy \vee \bar{x}yz \vee \bar{w},\quad x \vee y \vee z,\quad xyz,\quad \bar{w},\quad 1.$$

Последние три выражения представляют собой частные случаи ДНФ, состоящей из одной элементарной конъюнкции.

Рассмотрим процедуру приведения произвольной булевой формулы к виду ДНФ. Булевы формулы представляются цепочками, составленными из операций $\vee$, $\wedge$ и $\neg$, символов переменных и скобок, определяющих порядок выполнения операций. Для преобразования булевой формулы к виду ДНФ выполняются следующие действия.

1. Все отрицания «спускаются» до переменных. Это производится с помощью правил двойного отрицания и де Моргана:

$$\bar{\bar{x}} = x;$$
$$\overline{x \wedge y} = \bar{x} \vee \bar{y};$$
$$\overline{x \vee y} = \bar{x} \wedge \bar{y}.$$

В результате символы отрицания будут присутствовать в формуле только над переменными.

2. Скобки раскрываются по закону дистрибутивности конъюнкции относительно дизъюнкции:

$$x \wedge (y \vee z) = xy \vee xz.$$

В результате формула будет приведена к виду ДНФ. Далее могут быть проведены упрощения ДНФ с использованием основных законов булевой алгебры и формул простого склеивания, поглощения, обобщенного склеивания и др. Например, приведем к виду ДНФ и упростим следующую булеву формулу:

$$\bar{x} \vee \overline{xy}\,\overline{x \vee y} \vee xy(x \vee y) \vee \bar{z} = \bar{x} \vee (\bar{x} \vee \bar{y})\,\bar{x}\,\bar{y} \vee xyx \vee xyy \vee \bar{z} =$$

$$= \bar{x} \vee \bar{x}\,\bar{y} \vee xy \vee \bar{z} = \bar{x}(1 \vee \bar{y}) \vee xy \vee \bar{z} = \bar{x} \vee xy \vee \bar{z} = x \vee y \vee \bar{z}.$$

Более общий прием снятия знаков отрицания перед скобками основан на использовании правила Шеннона, вытекающего из принципа двойственности (см. пункт 6.3.2).

Правило Шеннона: если в булевой формуле $F = \bar{f}(X)$ все знаки операций заменить на двойственные (конъюнкцию на дизъюнкцию, дизъюнкцию на конъюнкцию), а все переменные инверсировать, то полученная формула будет равносильна формуле F.

Например:

$$F = \overline{\bar{x}y(xz \vee \bar{y}w)} = x \vee \bar{y} \vee (\bar{x} \vee \bar{z})(y \vee \bar{w}).$$

5.4.2 Совершенная дизъюнктивная нормальная форма

Элементарная конъюнкция называется *полной* относительно переменных $x_1, x_2, \ldots, x_n$, если она содержит символы всех переменных (может быть под знаками отрицания). Ранг полной конъюнкции равен n.

Например, конъюнкция $\bar{x}\,\bar{y}\,z$ полна относительно трех переменных x, y, z, но не полна относительно четырех переменных: w, x, y, z.

Полная элементарная конъюнкция в общем виде может быть представлена как $k = x_1^{\sigma_1} x_2^{\sigma_2} \ldots x_n^{\sigma_n}$, где $\sigma_i \in \{0, 1\}$ и $x^{\sigma} = x\,\sigma \vee \bar{x}\,\bar{\sigma}$, n – число переменных. Полная принимает значение 1 на единственном наборе значений ее переменных: $\boldsymbol{\sigma} = \sigma_1\,\sigma_2 \ldots \sigma_n$. В силу этого элементарную полную конъюнкцию называют *конституентой единицы* (в литературе используется также термин *минтерм*) относительно переменных $x_1, x_2, \ldots, x_n$ (если она полна относительно этого множества переменных). Каждой конституенте единицы соответствует единственный набор $\boldsymbol{\sigma}$ значений переменных $x_1, x_2, \ldots, x_n$, на котором она принимает значение 1, на остальных наборах она принимает значение 0.

Например, $\bar{x}_1 x_2 x_3 \bar{x}_4$ принимает значение 1 только на наборе 0110.

Очевидно, что конъюнкция любого числа различных конституент единицы равна нулю, поскольку не существует ни одного набора значений аргументов, на которых хотя бы две полные элементарные конъюнкции принимали значение 1.

Частным случаем ДНФ является *совершенная дизъюнктивная нормальная форма* (СДНФ). СДНФ представляет собой многоместную дизъюнкцию полных элементарных конъюнкций.

Для произвольной булевой функции, заданной в табличной форме или определенной характеристическим множеством M_f^1, легко построить представляющую ее СДНФ. Для этого достаточно выделить наборы $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ значений переменных, на которых функция принимает значение 1, и записать каждый из них в виде полной элементарной конъюнкции, которая содержит положительные литералы тех переменных, которые принимают в данном наборе значение 1, и отрицательные литералы – в противном случае.

Например, СДНФ для функции от трех аргументов, заданной таблицей 5.9, имеет следующий вид:

$$f = \bar{x}\,y\,\bar{z} \vee x\,\bar{y}\,\bar{z} \vee x\,\bar{y}\,z\,.$$

И наоборот, по произвольной СДНФ, определенной на переменных $x_1, x_2, \ldots, x_n$ и состоящей из m конституент единицы, можно однозначно построить характеристическое множество M_f^1 булевой функции $f(x_1, x_2, \ldots, x_n)$, которое будет состоять из m

наборов значений переменных, соотвествующих конституентам единицы этой СДНФ.

Таблица 5.9 – Трехместная функция $f(x, y, z)$

x	y	z	$f(x, y, z)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Существует взаимно однозначное соответствие между булевой функцией $f(x_1, x_2, \ldots, x_n)$ и ее СДНФ, и, следовательно, для любой булевой функции, не являющейся тождественно равной нулю, существует единственная СДНФ (с точностью до порядка конъюнкций в СДНФ и литералов в конъюнкциях). Поэтому СДНФ является *канонической формой* представления булевой функции.

Константа 1 представляется в виде СДНФ, которая содержит все различные элементарные конъюнкции, полные относительно рассматриваемого множества переменных.

Рассмотрим преобразование произвольной булевой формулы к виду СДНФ. Сначала булева формула преобразуется к виду ДНФ, затем каждая неполная относительно всех переменных элементарная конъюнкция заменяется на ДНФ, членами которой являются полные конъюнкции. Если в конъюнкцию $k = x_1^{\sigma 1} x_2^{\sigma 2} \ldots x_m^{\sigma m}$ не входит переменная y, содержащаяся в других конъюнкциях, то k заменяется на

$$x_1^{\sigma_1} x_2^{\sigma_2} \ldots x_m^{\sigma_m} (y \vee \bar{y}) = x_1^{\sigma_1} x_2^{\sigma_2} \ldots x_m^{\sigma_m} y \vee x_1^{\sigma_1} x_2^{\sigma_2} \ldots x_m^{\sigma_m} \bar{y}.$$

Например:

$$x\,\bar{y} \vee \bar{x} = x\,\bar{y}(z \vee \bar{z}) \vee \bar{x}(y \vee \bar{y})(z \vee \bar{z}) = x\,\bar{y}z \vee x\,\bar{y}\,\bar{z} \vee \bar{x}yz \vee \bar{x}\,\bar{y}z \vee \bar{x}y\,\bar{z} \vee \bar{x}\,\bar{y}\,\bar{z}.$$

При приведении к виду ДНФ и СДНФ произвольной формулы, которая содержит операции, отличные от булевых, она сначала преобразуется к булевому виду путем выражения этих операций через булевы.

Представление операций алгебры логики через булевы операции конъюнкции, дизъюнкции и отрицания можно легко получить из табличного задания соответствующих функций путем построения и упрощения СДНФ. Например, функции, заданные в таблице 5.10, представляются в следующем виде:

$$x \to y = \bar{x}\,\bar{y} \vee \bar{x}y \vee xy = \bar{x}\,\bar{y} \vee y = \bar{x} \vee y.$$

$$x \oplus y = \bar{x}y \vee x\,\bar{y};$$

$$x \sim y = \bar{x}\,\bar{y} \vee xy.$$

Таблица 5.10 – Некоторые функции алгебры логики

x	y	$x \to y$	$x \oplus y$	$x \sim y$
0	0	1	0	1
0	1	1	1	0
1	0	0	1	0
1	1	1	0	1

Приведем к виду ДНФ и упростим следующую формулу алгебры логики:

$$F = \bar{x} \oplus yz \to \bar{y} \vee (x \sim \bar{z}) = (\bar{x} \oplus yz) \to (\bar{y} \vee (x \sim \bar{z})) = (\bar{x} \sim yz) \vee (\bar{y} \vee (x \sim \bar{z})) =$$

$$= \bar{x}yz \vee x(\bar{y} \vee \bar{z}) \vee \bar{y} \vee \bar{x}z \vee x\,\bar{z} = \bar{x}yz \vee x\,\bar{y} \vee \bar{y} \vee \bar{x}z \vee x\,\bar{z} = \bar{y} \vee \bar{x}z \vee x\,\bar{z}.$$

При проведении преобразований кроме приведенных выше формул использовались также следующие равносильности:

$$\overline{x \oplus y} = x \sim y;$$

$$xy \vee y = y;$$

$$xy \vee \bar{y} = x \vee \bar{y}.$$

5.4.3 Конъюнктивная нормальная форма

Как уже отмечалось, конъюнкция и дизъюнкция двойственны друг другу. Соответственно, все сказанное выше об элементарных конъюнкциях и дизъюнктивных нормальных формах можно перефразировать на случай элементарных дизъюнкций и конъюнктивных нормальных форм.

Элементарной дизъюнкцией называется многоместная дизъюнкция попарно различных литералов. К числу элементарных дизъюнкций относятся также выражения, состоящие из одного литерала, а также константа нуль (0) как дизъюнкция пустого числа литералов. Число литералов элементарной дизъюнкции называется ее *рангом*.

По определению элементарными дизъюнкциями являются, например, выражения

$$0,\ \ x,\ \ \bar{x},\ \ x \vee y,\ \ x \vee \bar{y} \vee z,\ \ \bar{x} \vee \bar{y} \vee \bar{z},$$

имеющие ранги соответственно 0, 1, 1, 2, 3, 3. В то же время элементарными дизъюнкциями не являются следующие выражения:

$$1,\ \overline{x \vee y},\ \ x \vee y \vee x,\ \ \bar{x} \vee x.$$

Конъюнктивной нормальной формой (КНФ) называется формула, представляющая конъюнкцию элементарных дизъюнкций. Входящие в КНФ элементарные дизъюнкции называются также дизъюнктивными членами или *дизъюнктами.* По определению следующие выражения являются примерами КНФ:

$$(x \vee y)(\bar{x} \vee y \vee z \vee \bar{w}),\ \ xy,\ \ x \vee y \vee z,\ \ \bar{x},\ \ 0.$$

Последние четыре выражения представляют собой частные случаи КНФ, когда она состоит из двух одноранговых элементарных дизъюнкций, одной элементарной дизъюнкции или константы 0.

Представление функции в виде КНФ, так же как и в виде ДНФ, не единственно. Однако приведение формул к виду КНФ дает возможность проводить равносильные преобразования формул с использованием законов и равносильностей булевой алгебры.

Преобразование произвольной булевой формулы в КНФ может быть выполнено аналогично приведению формулы к ДНФ. Для «расщепления» элементарных конъюнкций ранга, большего 1, используется свойство дистрибутивности дизъюнкции относительно конъюнкции:

$$x \vee (yz) = (x \vee y)(x \vee z).$$

Например:

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

$$= xy \vee \bar{z} = (x \vee \bar{z})(y \vee \bar{z}).$$

5.4.4 Совершенная конъюнктивная нормальная форма

Элементарная дизъюнкция называется *полной* относительно переменных $x_1, x_2, \ldots, x_n$, если она содержит символы всех этих переменных (может быть под знаком отрицания). Ранг полной дизъюнкции равен n.

Например, дизъюнкция $\bar{x} \vee \bar{y} \vee z$ полна относительно трех переменных x, y, z, но не полна относительно четырех переменных w, x, y, z.

Полная элементарная дизъюнкция в общем виде может быть обозначена как $d = x_1^{\sigma_1} \vee x_2^{\sigma_2} \vee ... \vee x_n^{\sigma_n}$, где $\sigma_i \in \{0, 1\}$ и $x^{\sigma} = x\,\sigma \vee \bar{x}\,\bar{\sigma}$. Полная элементарная дизъюнкция принимает значение 0 на единственном наборе значений ее переменных: $\bar{\boldsymbol{\sigma}} = (\bar{\sigma}_1\ \bar{\sigma}_2 \ldots\ \bar{\sigma}_n)$, на остальных наборах она принимает значение 1. В силу этого элементарную дизъюнкцию d называют *конституентой нуля* (в литературе используется также термин *макстерм*) относительно переменных $x_1, x_2, \ldots, x_n$ (если она полна относительно этого множества переменных).

Например, полная элементарная дизъюнкция $\bar{x}_1 \vee x_2 \vee x_3 \vee \bar{x}_4$ принимает значение 0 только на наборе 1 0 0 1.

Очевидно, что дизъюнкция любого числа различных конституент нуля функции равна 1.

Частным случаем КНФ является *совершенная конъюнктивная нормальная форма* (СКНФ). СКНФ представляет собой многоместную конъюнкцию полных элементарных дизъюнкций.

Рассмотрим вопрос построения СКНФ для произвольной булевой функции $f(x_1, x_2, \ldots, x_n)$, не являющейся тождественно равной единице. Как это показано выше, по табличному представлению булевой функции можно легко построить представляющую ее СДНФ. В силу же двойственности ДНФ и КНФ можно перейти от СДНФ к виду СКНФ. Представим функцию $g = \bar{f}(x_1, x_2, \ldots, x_n)$ в виде СДНФ:

$$g = \bigvee_{i=0}^{2^n-1} x_1^{\sigma_{i1}} x_2^{\sigma_{i2}} \ldots x_n^{\sigma_{in}} g_i,$$

где коэффициенты $g_i = g(\sigma_{i1}, \sigma_{i2}, \ldots, \sigma_{in})$ имеют значение либо 0, либо 1 в зависимости от того, какое значение функция g принимает на i-м наборе. Возьмем двухстороннее отрицание приведенной формулы:

$$\bar{g} = \neg(\bigvee_{i=0}^{2^n-1} x_1^{\sigma_{i1}} x_2^{\sigma_{i2}} \ldots x_n^{\sigma_{in}} g_i) = \bigwedge_{i=0}^{2^n-1} (x_1^{\bar{\sigma}_{i1}} \vee x_2^{\bar{\sigma}_{i2}} \vee \ldots \vee x_n^{\bar{\sigma}_{in}} \vee \bar{g}_i).$$

Так как $g = \bar{f}$, то $\bar{g}_i = f_i$, где $f_i = f(\sigma_{i_1}, \sigma_{i_2}, \ldots, \sigma_{i_n})$ – значение функции $f(x_1, x_2, \ldots, x_n)$ на i-м наборе значений ее аргументов. Исходя из этого, справедливо

$$f = \bar{g} = \bigwedge_{i=0}^{2^n-1} (x_1^{\bar{\sigma}_{i1}} \vee x_2^{\bar{\sigma}_{i2}} \vee \ldots \vee x_n^{\bar{\sigma}_{in}} \vee f_i). \tag{5.3}$$

Для наборов $(\sigma_{i_1}, \sigma_{i_2}, \dots, \sigma_{i_n})$, обращающих $f_i = f(\sigma_{i_1}, \sigma_{i_2}, \dots, \sigma_{i_n})$ в единицу, соответствующий сомножитель $(x_1^{\bar{\sigma}_{i_1}} \vee x_2^{\bar{\sigma}_{i_2}} \vee \dots \vee x_n^{\bar{\sigma}_{i_n}} \vee f_i)$ в формуле (5.3) равен 1. Следовательно, конъюнкцию в правой части выражения (5.3) нужно брать только по тем наборам $\sigma_{i_1}, \sigma_{i_2}, \dots, \sigma_{i_n}$, для которых $f_i = 0$, т. е. булева функция может быть представлена в виде СКНФ согласно формуле

$$f(x_1, x_2, \dots, x_n) = \bigwedge_{f(\sigma_{i_1}, \sigma_{i_2}, \dots, \sigma_{i_n}) = 0} (x_1^{\bar{\sigma}_{i_1}} \vee x_2^{\bar{\sigma}_{i_2}} \vee \dots \vee x_n^{\bar{\sigma}_{i_n}}).$$

Из этой формулы следует, что СКНФ любой булевой функции f содержит ровно столько конституент нуля, сколько наборов значений аргументов содержится в ее области M_f^0 нулевых значений.

При построении СКНФ функции f выделяются те наборы значений аргументов, на которых она принимает значение 0, и для каждого такого набора в СКНФ вводится полная элементарная дизъюнкция, содержащая положительные литералы тех переменных, которые имеют в этом наборе значение 0, и отрицательные – в противном случае (т. е. значения переменных в этих наборах инвертируются).

Например, если $f(x_1, x_2, x_3)$ принимает значение 0 на наборе 0 1 0, то в СКНФ вводится полная элементарная дизъюнкция $x_1 \vee \bar{x}_2 \vee x_3$.

Константа 0 представляется в виде СКНФ, которая содержит все различные элементарные дизъюнкции, полные относительно рассматриваемого множества переменных.

СКНФ для функции от трех аргументов, заданной таблицей 5.9, имеет следующий вид:

$$f = (x_1 \vee x_2 \vee x_3)(x_1 \vee x_2 \vee \bar{x}_3)(x_1 \vee \bar{x}_2 \vee \bar{x}_3)(\bar{x}_1 \vee \bar{x}_2 \vee x_3)(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3).$$

Очевидно, что для любой булевой функции $f(x_1, x_2, \dots, x_n)$, не являющейся тождественно равной единице, существует единственная СКНФ (с точностью до порядка элементарных дизъюнкций в СКНФ и литералов в дизъюнкциях). Поэтому данная форма представления булевой функции является *канонической*, так же как и СДНФ.

Для приведения произвольной булевой формулы к виду СКНФ сначала она преобразуется к виду КНФ. Затем каждая неполная относительно всех переменных элементарная дизъюнкция заменяется на КНФ полных элементарных дизъюнкций. Для этого последовательно выбираются переменные, которые в нее не входят, но содержатся в других дизъюнкциях, и за счет них расширяется данная дизъюнкция. Например, если в дизъюнкцию $x_1^{\sigma_1} \vee x_2^{\sigma_2} \vee \ldots \vee x_m^{\sigma_m}$ не входит переменная y, то дизъюнкция заменяется выражением

$$x_1^{\sigma_1} \vee x_2^{\sigma_2} \vee \ldots \vee x_m^{\sigma_m} \vee \bar{y}y = (x_1^{\sigma_1} \vee x_2^{\sigma_2} \vee \ldots \vee x_m^{\sigma_m} \vee \bar{y})\,(x_1^{\sigma_1} \vee x_2^{\sigma_2} \vee \ldots \vee x_m^{\sigma_m} \vee y).$$

Аналогично это выполняется и для КНФ:

$$(w \vee x \vee \bar{z})\,(\bar{w} \vee \bar{y}) = (w \vee x \vee \bar{z} \vee \bar{y}y)\,(\bar{w} \vee \bar{y} \vee \bar{x}x \vee \bar{z}z) =$$

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

$$\wedge\,(\bar{w} \vee \bar{y} \vee x \vee \bar{z})\,(\bar{w} \vee \bar{y} \vee x \vee z).$$

5.4.5 Связь ДНФ и КНФ, взаимные преобразования

Для n переменных и любой булевой функции f от этих переменных нетрудно доказать следующие утверждения, вытекающие из приведенных выше утверждений и определений.

1. Существует 2^n различных конституент единицы.
2. Существует 2^n различных конституент нуля.
3. Дизъюнкция всех конституент единицы равна 1.
4. Конъюнкция всех конституент нуля равна 0.
5. Конъюнкция любых двух конституент единицы равна 0.
6. Дизъюнкция любых двух конституент нуля равна 1.
7. Отрицание функции f равно дизъюнкции тех и только тех конституент единицы, которые не входят в СДНФ функции f.
8. Отрицание функции f равно конъюнкции тех и только тех конституент нуля, которые не входят в СКНФ функции f.
9. Для любой функции f суммарное число конституент единицы и нуля (входящих соответственно в ее СДНФ и СКНФ) равно 2^n.
10. Инверсией конституенты единицы является конституента нуля, и наоборот:

$$\bar{k}_p = d_q; \quad \bar{d}_q = k_p,$$

где двоичная запись числа q образуется из двоичной записи числа p путем замены единиц на нули, а нулей – на единицы.

11. Если функция f представлена в одной из совершенных нормальных форм (СДНФ или СКНФ), то можно получить ее отрицание $\bar{f}$ следующим образом:

– всюду поменять знак $\wedge$ на $\vee$ и наоборот;

– переменные, входящие в нормальную форму функции f без отрицания, взять с отрицанием;

– переменные, входящие в нормальную форму функции f с отрицанием, взять без отрицания.

На основании утверждений 7 и 10 укажем способ перехода от СДНФ к СКНФ некоторой функции. Для этого нужно:

1) образовать дизъюнкцию конституент единицы, не входящих в СДНФ;

2) символы конъюнкции заменить на символы дизъюнкции, и наоборот, и взять отрицания всех литералов.

Аналогичным образом осуществляется переход от СКНФ к СДНФ.

Продемонстрируем процедуру перехода от СДНФ к СКНФ на примере СДНФ функции, заданной таблицей 5.9:

$$f = k_2 \vee k_4 \vee k_5 = \bar{x} y \bar{z} \vee x \bar{y} \bar{z} \vee x \bar{y} z .$$

Найдем инверсию этой функции, используя утверждение 7:

$$\bar{f} = k_0 \vee k_1 \vee k_3 \vee k_6 \vee k_7 = \bar{x} \bar{y} \bar{z} \vee \bar{x} \bar{y} z \vee \bar{x} y z \vee x y \bar{z} \vee x y z .$$

Искомая СКНФ имеет вид

$$f = (x \vee y \vee z)(x \vee y \vee \bar{z})(x \vee \bar{y} \vee \bar{z})(\bar{x} \vee \bar{y} \vee z)(\bar{x} \vee \bar{y} \vee \bar{z}) = d_0 \wedge d_1 \wedge d_3 \wedge d_6 \wedge d_7 .$$

5.5 Логика высказываний и логический вывод

Логика высказываний (или пропозициональная логика) – это формальная теория, основным объектом которой служит понятие логического высказывания. Логика высказываний близка к логике неформальных рассуждений человека и известна еще со времен Античности.

5.5.1 Высказывания

Понятие *высказывания* лежит в основе любых рассуждений. В рамках логики высказываний рассматриваются значения истинности высказываний, а не их смысл, изучаются взаимоотношения между сложными высказываниями, образованными из простых. Простое высказывание является первичным в логике высказываний и не имеет строгого определения. Можно сказать, что высказывание представляет собой декларативное предложение (факт), которое или истинно, или ложно (не может быть истинным и ложным одновременно).

Например, высказываниями являются следующие предложения:

«Минск – столица Беларуси»;

«Собака – друг человека»;

«Летом тепло»;

«Снег холодный»;

«2 + 2 = 5».

Первые четыре высказывания истинны, последнее – ложно. Следующие предложения высказываниями не являются: «Ожидается ли дождь сегодня?», «Чему равно два плюс два?».

Если высказывание истинно, говорят, что оно принимает значение «истина», если оно ложно – значение «ложь».

Высказывание может быть *абсолютно истинным* или *абсолютно ложным*. Такое высказывание называется логической *константой*. Например, приведенные выше высказывания являются логическими константами.

Все рассмотренные выше высказывания (и переменные, и константы) являются *простыми* (неделимыми) высказываниями. Из простых высказываний с помощью логических *связок* (или операций) образуются *сложные* высказывания.

Например, следующее высказывание является сложным: «В кинотеатре предстоял показ фильма, и зрители заполнили зал». Оно образовано с помощью связки «и» из двух простых высказываний: «В кинотеатре предстоял показ фильма» и «Зрители заполнили зал». Связками являются также «или», «не», «если …, то …», «либо …, либо …», «если и только если».

Истинность сложного высказывания определяется истинностью составляющих его простых высказываний и операцией, которой они связаны. Следует отметить, что в исчислении высказываний рассматривается только значение истинности высказываний и не принимается во внимание их смысл (если только по нему не определяется истинность).

5.5.2 Алгебраические представления

Рассмотрим зависимость истинности сложного высказывания от истинности составляющих его простых высказываний. Логические связки можно интерпретировать как логические операции или функции, определенные на множестве {*и* (истина), *л* (ложь)} со значениями из этого же множества. Будем обозначать далее простые высказывания малыми буквами латинского алфавита.

Вводятся шесть логических операций. Результаты действия этих операций как функции логических переменных a и b представлены в таблице 5.11. Любая часть этой таблицы, представляющая отдельную логическую операцию, называется *таблицей истинности*. В ней перечислены всевозможные комбинации значений истинности высказываний a и b и соответствующие значения истинности сложного высказывания.

Таблица 5.11 – Результаты выполнения логических операций

a	b	$\neg a$	$a \wedge b$	$a \vee b$	$a \oplus b$	$a \sim b$	$a \to b$
л	*л*	*и*	*л*	*л*	*л*	*и*	*и*
л	*и*	*и*	*л*	*и*	*и*	*л*	*и*
и	*л*	*л*	*л*	*и*	*и*	*л*	*л*
и	*и*	*л*	*и*	*и*	*л*	*и*	*и*

1. Конъюнкция высказываний. Обозначается $a \wedge b$ (или $a \,\&\, b$, или $a\,b$). Высказывание $a \wedge b$ читается как «a и b» и истинно тогда и только тогда, когда оба высказывания истинны.

2. Дизъюнкция высказываний. Обозначается $a \vee b$ (или $a + b$). Высказывание $a \vee b$ читается как «a или b», оно истинно, если хотя бы одно из высказываний a и b истинно, и ложно, если оба высказывания ложны.

3. Отрицание высказывания. Обозначается $\neg a$ (или $\bar{a}$). Высказывание $\bar{a}$ читается как «не a» или «не верно, что a». Оно истинно, когда a ложно. Отрицание высказывания – одноместная операция, в отличие от остальных, которые являются двухместными.

4. Дизъюнкция с исключением. Обозначается $a \oplus b$. Высказывание $a \oplus b$ читается как «a либо b» или как «или a, или b». Оно истинно, когда истинно только одно из высказываний a или b, и ложно, когда оба высказывания истинны или оба ложны.

5. Эквиваленция. Обозначается $a \sim b$. Высказывание $a \sim b$ может быть прочитано следующим образом: «a равносильно b», «a, если и только если b», «a тогда и только тогда, когда b». Оно истинно тогда и только тогда, когда значения истинности высказываний a и b совпадают.

6. Импликация. Обозначается $a \to b$. Высказывание $a \to b$ читается как «если a, то b». Это высказывание ложно только в том случае, когда a истинно, а b ложно. Во всех остальных случаях оно истинно.

Из перечисленных операций в логике высказываний создаются композиции, выражаемые формулами. Например, $(a \wedge b) \oplus \neg a$ является композицией из трех операций. Формулы, состоящие из единственного простого высказывания, называются *простыми*. Индуктивное определение формулы расширяет набор простых формул за счет *составных* формул, которые могут быть образованы путем связывания простых формул в разном порядке и разными операциями. Формула задает в общем случае некоторое сложное высказывание. Значение формулы определяется истинностью задаваемого ею высказывания: формула истинна (или ложна), если это высказывание истинно (или ложно).

Формальное определение формулы логики высказываний, порядок выполнения операций в формулах, вычисление значений формул аналогичны тем, что введены в алгебре логики (см. подраздел 5.1).

5.5.3 Основные тавтологии логики высказываний

Отношения между формулами (равносильности и импликации), равносильные преобразования формул в исчислении высказываний определяются аналогично тому, как это делалось в алгебре логики (см. подраздел 5.2).

Формальная импликация $A \Rightarrow B$ тесно связана с причинно-следственным отношением между высказываниями, представленными формулами A и B. При этом A выступает в роли причины (гипотезы), а B – в роли следствия. Причинно-следственная связь между A и B может быть выражена предложениями: «Если A, то и B», «A является достаточным основанием для B», «B при условии выполнения A» и т. д.

Напомним, что формула, которая является истинной хотя бы при одном наборе значений переменных, является выполнимой (см. пункт 5.2.3). Формула, ложная при всех наборах значений переменных, называется противоречием (или невыполнимой, или тождественно ложной), а формула, истинная при всех наборах значений переменных, называется тавтологией (или общезначимой, или тождественно истинной).

Тавтологии логики высказываний составляют основу логических заключений, используемых, в частности, в математике. Рассмотрим основные тавтологии исчисления высказываний.

Закон тождества: $a \to a$ – всякое высказывание логически следует из самого себя.

Закон противоречия: $\neg (a \wedge \neg a)$ – всякое высказывание не может быть одновременно истинным и ложным. Согласно этому закону формула $a \wedge \bar{a}$ тождественно ложна.

Закон исключенного третьего: $a \vee \neg a$ – для всякого высказывания истинно или оно само, или его отрицание.

Закон двойного отрицания: $\neg \neg a \sim a$ – отрицание отрицания любого высказывания равносильно самому высказыванию.

Закон «истина из чего угодно»: $a \to (b \to a)$ – если a является истинным высказыванием, то оно следует (выводится) из любого высказывания b (истинного или ложного) или (что равносильно) формула $b \to a$ истинна.

Закон «из ложного что угодно»: $\neg a \to (a \to b)$ – если a является ложным высказыванием, то из a следует любое высказывание b (истинное или ложное) или (что равносильно) формула $a \to b$ истинна.

Закон modus ponens (латинское «modus ponendo ponens» – «путь утверждения утверждений»): $(a \wedge (a \to b)) \to b$ – если высказывание a истинно и истинно то, что из a следует b, то высказывание b также истинно. Эта тавтология используется в математических доказательствах и трактуется следующим образом: если все посылки верны, то и заключение также верно.

Закон modus tollens (латинское «modus tollendo tollens» – «путь исключения исключений»): $((a \to b) \wedge \neg b) \to \neg a$ – если из a следует b и b ложно, то a также ложно. Эта тавтология тоже используется в математических доказательствах. Такой способ доказательства называется «от противного» и трактуется следующим образом: высказывание a ложно, если из него выводится противоречие.

Закон силлогизма (греческое «syllogismos» – «вывод заключений»): $((a \to b) \wedge (b \to c)) \to (a \to c)$ – если из a следует b, а из b следует c, то из a следует c. Согласно этому закону при доказательствах утверждений в математике можно строить сколь угодно длинные цепочки заключений.

5.5.4 Логический вывод

Теоремой называется утверждение, истинность которого требует доказательства. Доказательство утверждения есть последовательность истинных высказываний, логически следующих друг из друга. Некоторые из этих высказываний истинны априори, другие могут быть аксиомами, ранее доказанными теоремами или гипотезами, истинность которых является посылкой теоремы. Наконец, некоторые высказывания логически выводятся из других в процессе доказательства.

Чтобы построить доказательство, необходимо иметь средство для вывода заключений – правила вывода. Эти правила позволяют сделать заключение из высказываний, которые являются или предполагаются истинными. Доказательство строится из гипотез, аксиом, ранее доказанных теорем с помощью правил вывода.

Частным случаем правил вывода являются *правила подстановки*. Они позволяют заменить любое высказывание на равносильное ему. Другие наиболее важные правила вывода задаются в форме тавтологий. Они могут быть легко преобразованы к виду правил формального вывода. Если тавтология включает импликацию, то тогда ее левая часть является *гипотезой*, а правая – *заключением*, которое логически вытекает из этой гипотезы.

Например, тавтология $(a \wedge (a \to b)) \to b$ (modus ponens) может быть записана как правило формального вывода: $(a, a \Rightarrow b) \Rightarrow b$ или

$$\frac{a,\ a \to b}{b}.$$

Это правило может быть интерпретировано как «Если известно, что высказывание a истинно и что из a следует b, то высказывание b также истинно» или «Если все посылки верны, то и заключение также верно».

Тавтология $((a \to b) \wedge \neg b) \to \neg a$ (modus tollens) порождает правило вывода: $(a \Rightarrow b, \neg b) \Rightarrow \neg a$ или

$$\frac{a \to b, \neg b}{\neg a}.$$

Это правило может быть интерпретировано как «Если известно, что истинно то, что из «a следует b» и то, что «b ложно», то «a ложно» также или «Высказывание a ложно, если из него выводится противоречие».

Тавтология $((a \to b) \wedge (b \to c)) \to (a \to c)$ (силлогизм) порождает правило вывода: $((a \Rightarrow b),(b \Rightarrow c)) \Rightarrow (a \Rightarrow c)$ или

$$\frac{a \to b, b \to c}{a \to c}.$$

Это правило может быть интерпретировано как «Если истинно то, что «из a следует b» и «из b следует c», то «из a следует c». Силлогизм состоит из трех высказываний: двух посылок и одного заключения, или представляет собой *дедуктивное* умозаключение, в котором на основании установления отношений меньшего и большего терминов по отношению к среднему термину в посылках устанавливается отношение между меньшим и большим терминами в заключении.

При доказательстве утверждений часто используются следующие очевидные правила вывода:

$$\frac{a \wedge b}{a,b};\ \frac{a,b}{a \wedge b};\ \frac{a \vee b, \neg a}{b};\ \frac{a}{a \vee b};\ \frac{a \to b}{a \vee c \to b \vee c};\ \frac{a \vee b, a \to c, b \to d}{c \vee d}.$$

Продемонстрируем процесс логического вывода на примере доказательства следующего утверждения:

«Если лошадь летает или корова ест мясо, то муха – птица. Если муха – птица, то соль сладкая. Но соль не сладкая. Следовательно, корова не ест мясо».

Первые три высказывания являются гипотезами, а последнее высказывание – заключением. Проверим, следует ли истинность заключения из истинности гипотез.

Выделим простые высказывания, фигурирующие в утверждении:

p – высказывание «лошадь летает»;

q – «корова ест мясо»;

r – «муха – птица»;

s – «соль сладкая».

Гипотезы представляются следующими сложными высказываниями:

1. $(p \vee q) \to r$.
2. $r \to s$.
3. $\bar{s}$.

Заключение: $\bar{q}$.

Таким образом, необходимо доказать, что высказывание, представляемое формулой

$$(((p \vee q) \to r) \wedge (r \to s) \wedge \bar{s}) \to \bar{q},$$

является тавтологией.

Один из возможных путей доказательства состоит в построении таблицы истинности для этого сложного высказывания. Если формула есть тавтология, тогда заключение логически следует из гипотез. Однако этот способ требует построения таблицы истинности, которая становится громоздкой при большом числе гипотез. Проще построить доказательство, используя гипотезы и правила вывода.

Ход доказательства продемонстрирован ниже. Слева показаны заключения, справа – порождающие его высказывания и правила вывода:

Заключение	Основание
1. $(p \vee q) \to r$	Гипотеза 1
2. $r \to s$	Гипотеза 2
3. $(p \vee q) \to s$	Шаги 1 и 2 и закон силлогизма
4. $\bar{s}$	Гипотеза 3
5. $\overline{p \vee q}$	Шаги 3 и 4 и закон modus tollens
6. $\bar{p} \wedge \bar{q}$	Шаг 5 и закон де Моргана: $\overline{x \vee y} = \bar{x}\ \bar{y}$
7. $\bar{q} \wedge \bar{p}$	Шаг 6 и коммутативность относительно $\wedge$
8. $\bar{q}$	Шаг 7 и закон упрощения $(p \wedge q) \to p$

Последнее высказывание является заключением, следующим из гипотез: если гипотезы истинны, то и заключение также истинно.

Приведем логический вывод еще одного утверждения:

«Если цены высоки, то зарплата тоже высока. Цены высоки или используется регулирование цен. Если применяется регулирование цен, то нет инфляции. Но инфляция наблюдается. Следовательно, зарплата высока».

Выделим простые высказывания, фигурирующие в утверждении:

p – высказывание «цены высоки»;

s – «зарплата высока»;

r – «используется регулирование цен»;

i – «применяется регулирование цен».

Гипотезы представляются следующими высказываниями:

1. $p \to s$;
2. $p \vee r$;
3. $r \to \bar{i}$;
4. i.

Заключение: s.

Доказательство следует из применения следующих правил вывода:

$$\frac{r \to \neg i,\, i}{\neg r};\quad \frac{p \vee r,\, \neg r}{p};\quad \frac{p,\, p \to s}{s}.$$

5.6 Логика предикатов

Логика предикатов представляет собой обобщение логики высказываний. Логика высказываний оперирует с высказываниями, простыми и сложными (составленными из простых). Простое высказывание представляет собой повествовательное предложение, которое может быть либо истинным, либо ложным. В соответствии с этим простые высказывания рассматриваются в логике высказываний как переменные, принимающие значения из множества $\{и, л\}$. В отличие от логики высказываний в логике предикатов переменные могут быть многозначными и даже бесконечнозначными.

5.6.1 Предикаты и кванторы

Основным понятием логики предикатов является *предикат* – логическая двузначная функция $P(x_1, x_2, \dots, x_n)$ от многозначных переменных $x_1, x_2, \dots, x_n$, называемых *предметными*. Множество возможных значений некоторого аргумента x_i предиката называется его *предметной областью* и обозначается M_i.

В зависимости от числа n аргументов предикаты называются *нуль-местными*, *одноместными*, *двухместными* и далее *n-местными*. Нуль-местные предикаты представляют собой простые высказывания.

Одноместный предикат $P(x)$, определенный на предметной области M, задает некоторое свойство, присущее или не присущее элементам из M, соответственно в алгебре высказываний предикат $P(x)$ интерпретируется как высказывание «x обладает свойством P», и это высказывание в зависимости от x может быть истинным или ложным.

Одноместному предикату $P(x)$ на множестве M можно поставить во взаимно однозначное соответствие унарное отношение или подмножество $M_x \subseteq M$ значений аргумента x, для которых $P(x)$ истинно: если $m_i \in M_x$, то $P(m_i) = u$.

Примеры одноместных предикатов:

«x – простое число» на множестве натуральных чисел;

«x – студент» на некотором множестве людей;

«x – ядовитый гриб» на множестве грибов.

Двухместный предикат $P(x, y)$ определяется на множестве M_P, которое является декартовым произведением областей возможных значений его аргументов $M_P = M_1 \times M_2$. Предикат $P(x, y)$ задает бинарное отношение между элементами из множеств M_1 и M_2, являющимися предметными областями для переменных x и y. В алгебре высказываний предикат $P(x, y)$ интерпретируется как высказывание «x находится в отношении P с y». При этом в множестве $M_1 \times M_2$ выделяется подмножество пар элементов $(m_{i1}, m_{i2}) \in M_1 \times M_2$, для которых $P(x, y)$ истинно.

Примеры двухместных предикатов:

«$x > y$» на множествах натуральных чисел;

«x женат на y» на множествах мужчин и женщин;

«x пересекается с y» на множествах прямых линий.

Предметная область n-местного предиката $P(x_1, x_2, \ldots, x_n)$ представляет собой декартово произведение областей возможных значений его аргументов:

$$M_P = M_1 \times M_2 \times \ldots \times M_n.$$

Некоторые из множеств $M_1, M_2, \ldots, M_n$ или все они могут совпадать. В последнем случае, когда все $M_i = M$, предметная область предиката представляет собой n-ю степень этого множества M^n.

Примеры трехместных предикатов:

«$x + y = z$» на множествах натуральных чисел (N^3);

«x – сын y и z» на множествах M_1 и M_2 мужчин и M_3 женщин;

«x проходит через y и z» на множестве M_1 прямых и множествах M_2, M_3 точек.

Всякий предикат при присвоении конкретных значений его предметным переменным превращается в простое высказывание. Например, одноместный предикат «x – четное число», предметной областью которого является множество натуральных чисел, при фиксации значения $x = 5$ превращается в высказывание «5 – четное число», имеющее значение *л* («ложь»). Или двухместный предикат «x больше y», предметной областью которого является множество натуральных чисел, при фиксации значений $x = 9$ и $y = 6$ превращается в высказывание «9 больше 6», имеющее значение *и* («истина»).

В общем случае существует взаимно однозначное соответствие между n-местными предикатами и n-арными отношениями. Всякому n-местному предикату $P(x_1, x_2, \ldots, x_n)$ можно поставить в соответствие n-арное отношение R, такое, что $(m_1, m_2, \ldots, m_n) \in R$, если и только если $P(m_1, m_2, \ldots, m_n) = и$, и любому n-арному отношению R можно поставить в соответствие предикат $P(x_1, x_2, \ldots, x_n)$, который определен на декартовом произведении $M_P = M_1 \times M_2 \times \ldots \times M_n$ и который принимает значение $P(m_1, m_2, \ldots, m_n) = и$, если и только если $(m_1, m_2, \ldots, m_n) \in R$.

Всякий n-местный предикат $P(x_1, x_2, \ldots, x_n)$ можно рассматривать как одноместный предикат $P(x)$ на множестве наборов

$$(m_1, m_2, \ldots, m_n) \in M_1 \times M_2 \times \ldots \times M_n .$$

Поскольку значением предиката является *и* или *л*, то к предикатам могут применяться все операции логики высказываний. Кроме того, в логике предикатов имеются специфические для них операции. Они выражают утверждения о всеобщности и существовании, применимые ко всей предметной области некоторой переменной предиката, и представляются посредством соответствующих кванторов *общности* ($\forall$) и *существования* ($\exists$).

Пусть $P(x)$ – некоторый одноместный предикат, принимающий значение *и* или *л* для каждого элемента из предметной области M.

Результат применения *квантора общности* $\forall$ (называемого также квантором всеобщности) к предикату $P(x)$ обозначается через $\forall x P(x)$ (иногда и через $xP(x)$) и представляет собой высказывание «для всех x из M имеет место $P(x)$». Таким образом, $\forall x P(x)$ представляет собой истинное высказывание, если $P(x)$ истинно для каждого элемента x из M: $\forall x P(x) = и$, если $P(x) = и$ для всех $x \in M$; $\forall x P(x) = л$ в противном случае.

Из определения выражения $\forall x P(x)$ следует, что для любого предиката $P(x)$ и любого элемента m из его предметной области M справедливо

$$\forall x P(x) \Rightarrow P(m), \text{ т. е. } (\forall x P(x) \to P(m)) = u.$$

Примеры использования квантора общности:

1. Утверждение «Всякое натуральное число является рациональным числом» запишется в виде $\forall x(N(x) \to R(x))$, где предикаты $N(x)$ и $R(x)$ следует читать как «x – натуральное число» и «x – рациональное число».

2. Утверждение «Студент сдал или не сдал экзамен» запишется в виде $\forall x(S(x) \to (P(x) \vee \overline{P}(x)))$, где через $S(x)$ и $P(x)$ обозначены высказывания «x – студент» и «x сдал экзамен».

Результат применения *квантора существования* $\exists$ к предикату $P(x)$ обозначается через $\exists x P(x)$ и представляет собой высказывание «существует x, для которого имеет место $P(x)$». Таким образом, $\exists x P(x)$ представляет собой истинное высказывание, если существует элемент x из M, для которого $P(x)$ истинно:

$\exists x P(x) = u$, если $P(x) = u$ хотя бы для одного $x \in M$;

$\exists x P(x) = л$ в противном случае.

Из определения выражения $\exists x P(x)$ следует, что для любого предиката $P(x)$ и любого элемента m из его предметной области M справедливо $P(m) \Rightarrow \exists x P(x)$, т. е. $P(m) \to \exists x P(x) = u$.

Примеры использования квантора существования:

1. Утверждение «Некоторые рациональные числа являются натуральными числами» запишется в виде $\exists x(R(x) \wedge N(x))$, где предикаты $N(x)$ и $R(x)$ означают «x – натуральное число» и «x – рациональное число».

2. «Существуют студенты, которые сдали экзамен» запишется в виде $\exists x(S(x) \wedge P(x))$, где через $S(x)$ и $P(x)$ представляют высказывания «x – студент» и «x сдал экзамен».

Квантор (общности или существования) берется по одной переменной, но может применяться к предикату, зависящему от любого числа переменных. Переменная, по которой берется квантор, называется *связанной* переменной, а все остальные – *свободными*.

Применение квантора к n-местному предикату по какой-нибудь переменной превращает его в $(n-1)$-местный предикат. Например, следующие выражения представляют собой $(n-1)$-местные предикаты:

$$\forall x_i P(x_1, x_2, \ldots, x_i, \ldots, x_n) = R(x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n);$$

$$\exists x_i P(x_1, x_2, \ldots, x_i, \ldots, x_n) = S(x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n).$$

В этих выражениях переменная x_i является связанной переменной, а все остальные – свободными.

Применение квантора общности к одноместному предикату $P(x)$ порождает тождественно истинное высказывание (константу u), если $P(x) = u$ на всей предметной области M предиката, и тождественно ложное высказывание (константу $л$) в противном случае. Применение квантора существования к одноместному предикату $P(x)$ порождает тождественно ложное высказывание (константу $л$), если $P(x) = л$ на всей предметной области M предиката, и тождественно истинное высказывание (константу u) в противном случае.

Квантор общности можно рассматривать как обобщение конъюнкции, а квантор существования – как обобщение дизъюнкции. Пусть предметная область M некоторого предиката $P(x)$ конечна и состоит из элементов $m_1, m_2, \ldots , m_n$. Тогда имеют место следующие соотношения:

$$\forall x P(x) = P(m_1) \wedge P(m_2) \wedge \ldots \wedge P(m_n);$$

$$\exists x P(x) = P(m_1) \vee P(m_2) \vee \ldots \vee P(m_n).$$

Например, формула логики предикатов $A = \forall x \exists y P(x,y)$, где $M_x = \{x_1, x_2\}$ и $M_y = \{y_1, y_2\}$ – области значений его аргументов, может быть представлена в виде формулы логики высказываний следующим образом:

$$A = \forall x((P(x,y_1) \vee P(x,y_2)) = (P(x_1,y_1) \vee P(x_1,y_2)) \wedge (P(x_2,y_1) \vee P(x_2,y_2)).$$

В случае бесконечной предметной области кванторы могут рассматриваться как конъюнкции и дизъюнкции с бесконечным числом членов.

5.6.2 Теоретико-множественная интерпретация предикатов

Пусть предикат $P(x)$ задан на предметной области M. Тогда ему можно поставить во взаимно однозначное соответствие подмножество M_P тех элементов $x^* \in M$, для которых значение $P(x^*)$ истинно:

$$P(x^*) = u \Leftrightarrow x^* \in M_P, \;\; P(x^*) = л \Leftrightarrow x^* \in M \setminus M_P. \qquad (5.4)$$

Аналогичным образом интерпретируется отрицание предиката $P(x)$:

$$\bar{P}(x^*) = u \Leftrightarrow x^* \in M \setminus M_P.$$

Из определения истинности (5.4) предиката следует, что для предикатов $P(x)$ и $Q(x)$ при любом значении x^* предметной переменной x очевидны соотношения

$$P(x^*) \vee Q(x^*) = u \Leftrightarrow x^* \in M_P \cup M_Q;$$

$$P(x^*) \wedge Q(x^*) = u \Leftrightarrow x^* \in M_P \cap M_Q.$$

Пусть n-местный предикат $P(x_1, x_2, \dots , x_n)$ задан на предметной области M^n и x_i* – некоторое значение предметной переменной x_i ($i = 1, 2, \dots , n$). Тогда предикату $P(x_1, x_2, \dots , x_n)$ можно поставить в соответствие подмножество M_P тех элементов $(x_1^*, x_2^*, \dots , x_n^*) \in M^n$, для которых значение $P(x_1, x_2, \dots , x_n)$ истинно. Таким образом, для любого набора значений $x_1^*, x_2^*, \dots , x_n^*$ переменных $x_1, x_2, \dots , x_n$ имеют место

$$P(x_1^*, x_2^*, \dots , x_n^*) = и \Leftrightarrow (x_1^*, x_2^*, \dots , x_n^*) \in M_P;$$

$$P(x_1^*, x_2^*, \dots , x_n^*) = л \Leftrightarrow (x_1^*, x_2^*, \dots , x_n^*) \in M^n \setminus M_P.$$

Рассмотрим теоретико-множественную интерпретацию кванторов. Пусть имеем $P(x) = \exists y Q(x, y)$, где $Q(x, y)$ – предикат с предметной областью M^2, $P(x)$ – предикат с предметной областью M. По аналогии с вышесказанным предикату $P(x)$ соответствует такое множество $M_P \subseteq M$, что $P(x^*) = и$ для любого $x^* \in M_P$. Последнее имеет место, если для данного x^* существует такое значение y^* переменной y, что $Q(x^*, y^*) = и$, т. е. пара (x^*, y^*) принадлежит множеству M_Q.

Элемент x^* в паре (x^*, y^*) является проекцией этой пары на x, что символически выражается как $x^* = пр_x(x^*, y^*)$. Проекцией $пр_xA$ множества $A \subseteq M_Q$ пар вида (x^*, y^*) на x является множество всех x^*, таких, что $пр_x(x^*, y^*) \in A$. Из определения проекции следует, что $M_P = пр_xM_Q$.

Квантор общности $\forall x P(x)$ может быть выражен через квантор существования:

$$\forall x\, P(x) = \overline{\exists} x\ \overline{P}(x),$$

тогда $P(x) = \forall y\, Q(x, y) = \overline{\overline{\exists}}\, x\ \overline{Q}(x, y)$.

Исходя из определения теоретико-множественной операции отрицания, имеем $M_P = M \setminus пр_x(M^2 \setminus M_Q)$.

5.6.3 Формулы логики предикатов

Предикатной переменной называется переменная, обозначаемая прописной буквой (например, P, Q, R), значениями которой являются предикаты. При необходимости (если это важно) число аргументов предикатов, соответствующих упоминаемой предикатной переменной, указывается либо с помощью верхнего числового индекса (например, P^2, Q^0, R^1), либо в скобках перечисляется необходимое число аргументов или пробелов (например, $P(x_1, x_2, x_3)$ или $P(\cdot,\cdot,\cdot)$).

Формальное определение понятия *формулы логики предикатов* дается путем указания множества исходных символов и способов их связывания знаками операций. В качестве исходных символов в определении используются:

1) предметные переменные, обозначаемые строчными буквами (например, x, y, z);

2) константы («*и*» и «*л*»);

3) предикатные переменные, обозначаемые прописными буквами (например, P, Q, R).

Переменные, задающие высказывания, особо не будем выделять, считая их частным случаем предикатов – нуль-местными предикатами (например, P или P^0).

В качестве операций в определении формулы используются символы:

1) логических операций из множества $\{\neg, \wedge, \vee, \oplus, \sim, \rightarrow\}$;

2) кванторов $\forall$ и $\exists$.

Понятие формулы логики предикатов определяется в два этапа.

1. Каждая нуль-местная предикатная переменная или константа считается формулой. Каждый отдельный предикат, во все места которого подставлены предметные переменные или константы, есть формула. Все переменные, входящие в предикат, считаются свободными.

2. Если A – формула логики предикатов, содержащая свободную переменную x, то $\forall xA$ и $\exists xA$ также являются формулами, где x – связанная переменная, а все остальные переменные имеют тот же характер, что и в A, т. е. связанные в A переменные остаются связанными, а свободные – свободными.

3. Если A и B – формулы, причем нет таких переменных, которые в одной из формул были бы свободными, а в другой – связанными, то формулами являются $\neg A$ и $(A * B)$, где «*» – любая операция из множества $\{\wedge, \vee, \oplus, \sim, \rightarrow\}$, а переменные в этих формулах имеют тот же характер, что и в формулах A и B.

4. Других формул в логике предикатов нет.

Всякая определенная таким образом формула представляет собой предикат от своих свободных переменных и не зависит от связанных переменных.

Областью (или зоной) действия квантора $\forall x$ или $\exists x$ в формулах $\forall x A$ или $\exists x A$ называется формула A. Если формула A, в свою очередь, содержит квантор, то последний называется подчиненным квантору, который действует на A. Квантор имеет *ранг* 1, если ему не подчинен ни один другой квантор, и ранг k, если максимальный ранг квантора, который ему подчинен, равен $k-1$. Максимальный ранг квантора, входящего в формулу, называется *кванторной глубиной* формулы.

Например, кванторная глубина формулы

$$T = \exists x\,(\forall y\, P(x, y, z) \to \exists y\, Q(x, y)) \vee R(w)$$

равна двум. Областью действия квантора $\exists x$ является формула $\forall y\, P(x, y, z) \to \exists y\, Q(x, y)$. Переменная x связана квантором $\exists x$, y – кванторами $\forall y$ и $\exists y$, а w и z – свободные переменные. Ни одна переменная не входит в формулу T (в соответствии с пунктом 3 определения формулы) одновременно в связанном и свободном виде. В целом вышеприведенная формула представляет собой двухместный предикат $T(w, z)$, зависящий от своих свободных переменных.

Из определения квантора следует, что значение предиката, представленного некоторой формулой, не изменится, если некоторую связанную переменную заменить любой другой буквой, не используемой для обозначения свободных переменных. Отсюда следует, что в формуле можно так переобозначить связанные переменные, чтобы все кванторы действовали на разные связанные переменные. Следует, однако, при этом сохранять предметные области, привязанные к месту аргумента в формуле предиката.

Например, при этом условии формулы

$$\exists x\,(\forall y\, P(x, y, z) \to \exists y\, Q(x, y)) \vee R(w);$$
$$\exists x\,(\forall y\, P(x, y, z) \to \exists v\, Q(x, v)) \vee R(w);$$
$$\exists x\,(\forall v\, P(x, v, z) \to \exists y\, Q(x, y)) \vee R(w);$$
$$\exists v\,(\forall y\, P(v, y, z) \to \exists y\, Q(v, y)) \vee R(w)$$

могут представлять один и тот же предикат.

Рассмотрим отношения между формулами. Предикаты P и Q называются *равными*, если они имеют одинаковые значения при всех значениях входящих в них переменных.

Формулы P и Q логики предикатов называются *равносильными* (эквивалентными), если они задают один и тот же предикат. Как и в логике высказываний, равносильные формулы взаимно заменяемы.

О равносильности формул можно говорить относительно некоторой предметной области (тогда эти формулы равносильны на этой области) или безотносительно к любой предметной области (тогда они равносильны на любой области).

Все равносильности, имеющие место в логике высказываний, имеют место и в логике предикатов. Кроме этих равносильностей в логике предикатов имеются равносильности, связанные с кванторами. Рассмотрим основные свойства кванторов.

1. Связь между кванторами существования и общности:

$$\forall x P(x) = \overline{\exists} x \overline{P}(x); \quad \exists x P(x) = \overline{\forall} x \overline{P}(x). \qquad (5.5)$$

Эти равносильности являются аналогами законов де Моргана логики высказываний. Кванторы общности и существования можно считать двойственными друг другу. Равносильности позволяют заменять квантор общности квантором существования, и наоборот, а также переносить знак отрицания с квантора на формулу, находящуюся в зоне его действия, заменяя квантор на двойственный:

$$\overline{\exists} x P(x) = \forall x \ \overline{P}(x); \quad \overline{\forall} x P(x) = \exists x \ \overline{P}(x).$$

2. Коммутативность кванторов (одного типа):

$$\forall x \forall y P(x, y) = \forall y \forall x P(x, y); \quad \exists x \exists y P(x, y) = \exists y \exists x P(x, y).$$

Возможность перестановки местами одноименных кванторов следует из коммутативности конъюнкции и дизъюнкции, обобщением которых являются соответственно кванторы общности и существования.

Изменение порядка следования разноименных кванторов в общем случае недопустимо.

Это утверждение становится очевидным из следующего простого примера. Пусть $M_1 \times M_2$ – область определения двухместного предиката $P(x, y)$, M_1 и M_2 – соответственно множества мужчин и женщин, состоящих в браке, а предикат $P(x, y)$ задает отношение «x женат на y». На множестве $M_1 \times M_2$ имеет место $\forall x \exists y P(x, y) = и$. При изменении же порядка следования кванторов получим, что $\exists y \forall x P(x, y) = л$. Последняя формула содержательно означает, что существует некая y, состоящая в браке со всеми x, что является ложью.

3. Дистрибутивность квантора общности относительно конъюнкции:

$$\forall x(P(x) \wedge Q(x)) = \forall x P(x) \wedge \forall x Q(x).$$

Дистрибутивность квантора общности относительно дизъюнкции в общем случае не имеет места:

$$\forall x(P(x) \vee Q(x)) \neq \forall x P(x) \vee \forall x Q(x), \qquad (5.6)$$

так как предикат слева имеет значение *и* тогда, когда для каждого из значений x истинен хотя бы один из предикатов $P(x)$ или $Q(x)$ (для одних x это может быть $P(x)$, для других $Q(x)$), а предикат справа имеет значение *и* только тогда, когда хотя бы один из предикатов $P(x)$ или $Q(x)$ истинен для всех x.

Справедливость (5.6) становится очевидной из следующего простого примера. Пусть предикаты $P(x)$ и $Q(x)$ представляют соответственно высказывания «x имеет карий цвет глаз» и «x имеет голубой цвет глаз». Тогда на множестве людей, имеющих карий или голубой цвет глаз, предикат $\forall x(P(x) \lor Q(x))$ всегда имеет значение истина, а $\forall xP(x) \lor \forall xQ(x)$ истинен только тогда, когда все люди имеют один цвет глаз (или карий, или голубой).

Легко доказать, что отношение неравенства в формуле (5.6) можно заменить на отношение импликации, следующим образом:

$$\forall xP(x) \lor \forall xQ(x) \Rightarrow \forall x(P(x) \lor Q(x)),$$

так как если предикат слева имеет значение *и*, то автоматически истинным становится и предикат справа, а обратное не всегда имеет место, так как предикат справа имеет значение истина и тогда, когда для одного значения x истинен только один из предикатов, например $P(x)$ (а $Q(x)$ ложен), а для другого значения x истинен другой предикат – $Q(x)$ (а $P(x)$ ложен).

4. Дистрибутивность квантора существования относительно дизъюнкции:

$$\exists x(P(x) \lor Q(x)) = \exists xP(x) \lor \exists xQ(x).$$

Дистрибутивность квантора существования относительно конъюнкции в общем случае не имеет места:

$$\exists x(P(x) \land Q(x)) \neq \exists xP(x) \land \exists xQ(x),$$

но легко доказывается, что

$$\exists x(P(x) \land Q(x)) \Rightarrow \exists xP(x) \land \exists xQ(x),$$

так как для истинности левой части должен существовать x, для которого и $P(x)$, и $Q(x)$ истинны, для истинности же правой части значения x, для которых $P(x)$ и $Q(x)$ истинны, могут быть разными.

5. Равносильности с относительной константой.

Если предикат, стоящий под знаком квантора $\forall x$ или $\exists x$, не зависит от x, то он является константой относительно этого квантора. Относительную константу Q можно вынести из-под знака квантора:

$$\forall x(P(x) \land Q) = \forall xP(x) \land Q, \quad \forall x(P(x) \lor Q) = \forall xP(x) \lor Q;$$

$$\exists x(P(x) \land Q) = \exists xP(x) \land Q, \quad \exists x(P(x) \lor Q) = \exists xP(x) \lor Q.$$

Эти равносильности говорят о том, что дистрибутивные законы кванторов (и общности, и существования) относительно и конъюнкции, и дизъюнкции имеют место для случая относительных констант.

5.6.4 Нормальные формы логики предикатов

Любая формула логики высказываний может быть преобразована в формулу, определенную на множестве операций конъюнкции, дизъюнкции и отрицания.

Формула логики предикатов, в которой из операций логики высказываний имеются только конъюнкции, дизъюнкции и отрицания, причем символ отрицания стоит только над предикатными переменными, называется *приведенной* (или почти нормальной) формулой. Можно показать, что последовательное применение вышеприведенных равносильностей и выражение всех операций через $\wedge$, $\vee$ и $\neg$ позволяют привести к приведенной форме любую формулу логики предикатов.

Для примера преобразуем к приведенной форме следующую формулу:

$$T = \overline{\exists} x(\forall y P(x, y, z) \to \exists w Q(x, w)) \vee \forall z \ \overline{\forall} v\,(R(z) \vee S(v)).$$

Используя (5.5), получаем

$$T = \forall x(\overline{\forall y P(x,y,z) \to \exists w Q(x,w)} \vee \forall z \exists v(\overline{R(z) \vee S(v)}) .$$

Используя соотношение $x \to y = \bar{x} \vee y$ и закон де Моргана, получаем

$$T = \forall x(\overline{\overline{\forall} y P(x,y,z) \vee \exists w Q(x,w))} \vee \forall z \exists v(\overline{R}(z) \vee \overline{S}(v)) =$$

$$= \forall x(\forall y P(x,y,z) \wedge \overline{\exists} w Q(x,w)) \vee \forall z \exists v(\overline{R}(z) \vee \overline{S}(v)) =$$

$$= \forall x(\forall y P(x,y,z) \wedge \forall w \overline{Q}(x,w)) \vee \forall z \exists v(\overline{R}(z) \vee \overline{S}(v)) .$$

Последний результат и есть приведенная форма заданной формулы T.

Эта формула может быть далее упрощена следующим образом. Ранее было сказано, что связанную переменную предиката можно обозначить другой буквой, не используемой для обозначения свободной переменной. Следовательно, связанную переменную z (являющуюся свободной в предикате $P(x, y, z)$) в последнем члене заменим на t с целью последующего вынесения связывающего ее квантора $\forall$ в левую часть, где будут собраны все кванторы. Отсюда

$$T = \forall x(\forall y P(x,y,z) \wedge \forall w \overline{Q}(x,w)) \vee \forall t \exists v(\overline{R}(t) \vee \overline{S}(v)) .$$

Используя равносильности с относительными константами, окончательно получаем следующую форму:

$$T = \forall x \forall y \forall w \forall t \exists v(P(x,y,z) \wedge \overline{Q}(x,w) \vee \overline{R}(t) \vee \overline{S}(v)) .$$

Полученная форма, в которой все кванторы собраны слева, а формула, к которой применяются кванторы, является бескванторной, называется *нормальной формой* логики предикатов.

Задания

1. Опустить лишние скобки в формуле
$G = (p \wedge q) \to ((q \wedge \bar{p}) \to (r \wedge q))$.

2. Найти суперпозицию функций и упростить:
$g_1(x, y, z) = f_3(f_2(x, z), f_7(f_4(y,z)))$;
$g_2(x, y, z) = f_5(f_6(f_3(y,z), f_2(z, f_1(x, y))), x)$;
$g_3(x, y, z) = g_1(g_3(g_2(x, z), y))$, если:
$f_1(x, y) = x \vee y$;
$f_2(x, y) = x \wedge y$;
$f_3(x, y) = x \oplus y$;
$f_4(x, y) = x \to y$;
$f_5(x, y) = x \sim y$;
$f_6(x, y) = x \mid y$;
$f_7(x, y) = x \mid y$;
$g_1(x, y) = x \vee y$;
$g_2(x, y) = x \wedge y$;
$g_3(x, y) = x \to y$.

3. Вычислить значения формул по таблице истинности:
$F_1 = b\ \bar{c} \sim b \vee a$;
$F_2 = a \sim b \to \bar{b} \vee a$;
$F_3 = a \to \bar{b} \sim \bar{c} \vee b$;
$F_4 = a \to \bar{b}\, c \vee \bar{a}\, b$;
$F_5 = a\, b \to a \vee \bar{b} \to b\, c$.

4. Вычислить значение формул по польской записи и по представлению в виде дерева:
$F_1 = d \vee \bar{e} \to \bar{b} \sim a\, b \vee \bar{c}\, e$ при $a = c = d = 0$, $b = e = 1$;
$F_2 = \overline{a\, c} \vee b \to \bar{d} \vee e \sim a\ \bar{c}$ при $a = b = 0$, $c = d = e = 1$;

$G_1 = (\overline{p} \oplus s)(q \vee r) \vee ((p \to q) \sim q\,s)$ при $p = q = 0,\ r = s = 1$;

$G_2 = (q \to \overline{p} \oplus s \vee r) \vee (p \to q \sim q\,s)$ при $p = 1,\ q = 0,\ r = 0,\ s = 1$;

$G_3 = \overline{p} \oplus s\,q \to r \vee ((p \to q) \sim q\,s)$ при $p = q = 0 = r = 0,\ s = 1$;

$G_4 = a \oplus b \to (b \to c) \wedge (d \vee b\,c\,a)$ при $a = b = 0,\ c = 1$;

$G_5 = a \to \overline{b} \to c \vee a\,\overline{d}c \oplus (\overline{a} \oplus b)$ при $a = c = d = 0,\ b = 1$.

5. Проверить, находятся ли следующие формулы в одном из отношений: формальной импликации или равносильности:

$(a \to b)(c \to b)$ и $(a \vee c) \to b$;

$x\,y \vee \overline{x}z \vee yz$ и $x\,y \vee \overline{x}z$;

$(x \to y) \sim (z \oplus x)$ и $z \oplus xy$;

$xy \vee zt$ и $(x \vee z)(y \vee z)(x \vee t)(y \vee t)$;

$x\,\overline{y} \vee \overline{x}y \vee xz$ и $(x \sim y) \to xz$;

$xy \vee xz \vee \overline{y}\,\overline{z}$ и $\overline{x}\,\overline{z} \vee yz \vee x\,\overline{y}$;

$\overline{x}(\overline{y} \vee \overline{z}) \vee yz \vee x\,\overline{z}$ и $\overline{x} \vee y \vee \overline{z}$.

6. Выполнимы ли следующие формулы? Являются ли тавтологиями, противоречиями?

$a \wedge b \to a$;

$a \wedge b \to a \vee b$;

$(x \to y) \sim (\overline{y} \to \overline{x})$;

$(a \to b) \wedge (c \to b) \sim (a \vee c \to b \vee c)$;

$a\,(a \vee b \to b) \to b$;

$(a \to b)\,(c \to b) \sim (a \vee c \to b)$.

7. Упростить формулы:

$a\,b \vee \overline{c}\,e \sim (d \vee \overline{e} \to b)$;

$a \vee (\overline{a}\,\overline{b \vee c}) \sim (a \vee \overline{c \oplus b})$;

$a \vee (\overline{a}\,\overline{b \vee c}) \to (a \vee \overline{c \to b} \sim b)$;

$a\,(a\,b \vee \neg(b\,c \vee \neg(b \vee \overline{d}\,c)))$;

$(a \to b) \oplus (b \to ac) \vee (a \to c)$;

$a \vee \overline{a} \wedge \neg(b \vee c) \sim a \vee \neg(b \oplus c)\,c$;

$a \oplus b \oplus b \vee d \to \overline{b}\,d$;

$d \to b \oplus \overline{b} \sim c \oplus d$.

8. В чем различие между:

– ДНФ и совершенной ДНФ?

– КНФ и совершенной КНФ?

– конституентой нуля и конституентой единицы?

9. Какие из следующих конъюнкций являются элементарными конъюнкциями и полными элементарными конъюнкциями:

$$1, \quad x, \quad xy, \quad xyx, \quad x\bar{y}z, \quad \bar{x}x, \quad \bar{x}\,\bar{y}\,\bar{z}, \quad \overline{x\,y}.$$

10. Какие из следующих дизъюнкций являются элементарными дизъюнкциями и полными элементарными дизъюнкциями:

$$0, \quad x, \quad \bar{x} \vee x, \quad x \vee y, \quad x \vee \bar{y} \vee z, \quad \bar{x} \vee \bar{y} \vee \bar{z}, \quad x \vee \bar{y} \vee x, \quad \overline{x \vee y}.$$

11. Установить, какие из следующих ДНФ не являются совершенными ДНФ на множестве переменных $\{x, y, z, w\}$, и привести к виду совершенных ДНФ:

$xy \vee \bar{x}yz\bar{w}$;

$xyzw \vee \bar{x}z\bar{w} \vee xy\bar{z}w$;

$\bar{x}y\bar{z}\bar{w} \vee \bar{x}yz\bar{w} \vee x\bar{y}z\bar{w}$;

$xy \vee \bar{x}yz \vee \bar{w}$;

$x \vee y \vee z$;

$xyz\bar{w}$;

xyz.

12. Установить, какие из следующих КНФ не являются совершенными КНФ на множестве переменных $\{x, y, z, w\}$, и привести к виду совершенных КНФ:

$(x \vee y)(\bar{x} \vee y \vee z \vee \bar{w})$;

xyz;

$(x \vee y \vee z)(y \vee z \vee \bar{w})(\bar{z} \vee w)(\bar{x} \vee y \vee \bar{z} \vee \bar{w})$;

$x \vee y \vee z \vee \bar{w}$;

$\bar{x} \vee y \vee z$.

13. Какие из следующих выражений являются конституентами нуля, какие – конституентами единицы на множестве переменных $\{x, y, z, w\}$:

$xyz\bar{w}$;

$x \vee y \vee z$;

$xz\bar{w}$;

$x \vee \bar{y} \vee z \vee w$.

14. Представить операции «импликация» и «эквиваленция» в виде ДНФ и КНФ.

15. Представить в виде ДНФ и КНФ следующие булевы функции, заданные в векторном виде:

1 0 0 1 0 1 1 0;

0 1 0 1 0 1 0 1;

1 1 1 1 0 1 0 1.

16. Представить в виде ДНФ и КНФ следующие булевы функции, заданные таблицей истинности:

x	y	z	$f(x,y,z)$	$g(x,y,z)$
0	0	0	1	0
0	0	1	0	1
0	1	0	1	1
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

17. Перейти от функций, заданных в ДНФ, к эквивалентным представлениям в виде КНФ:

$f(x,y,z) = \bar{x}\,\bar{y}\,\bar{z} \vee \bar{x}\,\bar{y}z \vee x\bar{y}\,\bar{z} \vee xyz;$

$g(x,y,z) = \bar{x}y\bar{z} \vee \bar{x}yz \vee xy\bar{z} \vee xyz.$

18. Перейти от функций, заданных в КНФ, к эквивалентным представлениям в виде ДНФ:

$f(x,y,z) = (\bar{x} \vee \bar{y} \vee z)(\bar{x} \vee y \vee \bar{z})(x \vee \bar{y} \vee \bar{z})(\bar{x} \vee \bar{y} \vee z);$

$g(x,y,z) = (\bar{x} \vee \bar{y} \vee \bar{z})(\bar{x} \vee y \vee z)(x \vee \bar{y} \vee \bar{z})(x \vee \bar{y} \vee z)(x \vee \bar{y} \vee \bar{z}).$

19. Преобразовать булевы формулы к виду ДНФ (и к КНФ) и упростить:

$\bar{x} \vee \overline{x\,y}\;\overline{x \vee y} \vee xy(x \vee y) \vee \bar{z}.$

20. Построить ДНФ, таблицу истинности и двойственную функцию для:

– мажоритарной функции от трех аргументов (функция принимает значение 1 в наборах, в которых число единичных компонент больше числа нулевых);

– функции $(x \to y) \vee (x \oplus z)$.

21. Даны формулы логики высказываний. Проверить, выполнимы ли они, являются ли противоречиями или тавтологиями (доказать):

$a\,(a \vee b \to b) \to b;$

$(p \wedge (p \to q)) \to q;$

$(p \wedge (p \to q)) \to q;$

$((p \to q) \wedge \bar{q}) \to \bar{p};$

$((p \to q) \wedge (q \to r)) \to (p \to r);$

$(p \to q) \to (q \to p);$

$(p \to q) \vee (p \to (q \wedge p));$

$(p \wedge (q \to p) \to \bar{p});$

$((p \wedge \bar{q}) \to q) \to (p \to q);$

$(\bar{q} \to \bar{p}) \to ((\bar{q} \to p) \to q));$

$(q \to r) \to ((p \vee q) \to (p \vee r));$

$a \to a \vee b;$

$a \to \bar{\bar{a}};$

$(a \to b) \to (\bar{b} \to \bar{a}).$

22. Доказать, используя правила вывода:

1) «Если я завтра пойду на первую лекцию, то должен буду встать рано. А если я пойду вечером на дискотеку, то лягу поздно. Если я лягу поздно, а встану рано, то я вынужден буду довольствоваться только пятью часами сна. Но я просто не в состоянии обойтись пятью часами сна. Следовательно, я должен или пропустить завтра первую лекцию или не ходить на дискотеку»;

2) «Если курс математики интересен, то он полезен. Курс математики бесполезен или нетруден. Курс математики труден. Следовательно, этот курс неинтересен».

23. Исследовать системы высказываний на противоречивость, используя правила вывода:

1) $A \to C$, $\bar{C} \vee D$, $\bar{A} \to D$, $\bar{D}$;

2) $A \to \neg(BC)$, $D \vee E \to G$, $G \to \neg(H \vee I)$, $\bar{C}EH$.

24. Вывести заключения с указанием используемых законов (тождеств логики высказываний):

$$\frac{p \to q,\ r \to \bar{s},\ q \to r,\ s \vee \bar{q}}{\bar{p}};$$

$$\frac{\bar{a},\ b \to \bar{c},\ \bar{a} \to b,\ d \to c}{\bar{d}};$$

$$\frac{a \to c,\ c,\ \bar{a} \to d,\ d \vee \bar{e}}{\bar{e}}.$$

25. Прочитать выражения, указать зоны действия кванторов, перечислить связанные и свободные предметные переменные формул:

$\exists x(E(x) \wedge P(x)) \wedge \bar{\exists} x\{[E(x) \wedge P(x)] \wedge \exists y[(x \neq y) \wedge E(y) \wedge P(y)]\}$, где предикат $P(x)$ обозначает «x – простое число», $E(x)$ – «x – четное число»;

$\forall x\ \forall y\ (P(x) \wedge P(y) \wedge R(x,y) \to (x = y))$, где предикат $P(x)$ обозначает «x – простое число», $R(x,y)$ – «x делится на y»;

$\forall x\,(\bar{\exists} y[E(y,z) \to Q(x,y)])$;

$\forall x\,(P(x) \to \exists y\,(S(y,z) \vee Q))$.

26. Пусть D_1 и D_2 – множества деталей d_i двух механизмов, предикат $P(d_i)$ обозначает «деталь d_i – выполнена из чугуна». Записать на языке логики предикатов следующие высказывания:

– «Все детали первого механизма выполнены из чугуна»;

– «Детали, входящие в оба механизма выполнены из чугуна»;

– «Во втором механизме нет деталей из чугуна»;

– «Хотя бы одна деталь, входящая в какой-нибудь механизм выполнена из чугуна».

27. Дан предикат $P(x,y)$, где x и y – студенты из одной группы: $P(x,y) = u$, если x и y являются друзьями. Выразить следующие высказывания формулами логики предикатов:

– «Каждый студент имеет хотя бы одного друга в своей группе»;

– «По крайней мере один студент не имеет друзей в группе»;

– «По крайней мере один студент является другом для всех студентов группы».

28. Выразить формулами логики предикатов:

– «Каждый человек имеет хотя бы одного друга»;

– «Существуют люди, не имеющие друзей»;

– «Число а называется пределом числовой последовательности $\{x_1, x_2, \ldots, x_n, \ldots\}$, если для любого $\varepsilon > 0$ существует номер n_ε, зависящий от ε, такой что для любого $n > n_\varepsilon$ выполняется неравенство $|x_n - a| < \varepsilon$»;

– «Функция $f\,(x_1, x_2, \ldots, x_n)$ непрерывна в точке $M'(x_1', x_2', \ldots, x_n')$, если по любому $\varepsilon > 0$ найдется такое $\delta > 0$, что

$|f\,(x_1, x_2, \ldots, x_n) - f\,(x_1', x_2', \ldots, x_n')| < \varepsilon$ лишь только $|x_1 - x_1'| < \delta, \ldots, |x_n - x_n'| < \delta$ ».

29. Снять отрицания над квантором:

$$\forall x(\overline{\forall y U(x,y) \to \exists z V(x,z)}).$$

30. Выразить следующие формулы логики предикатов в виде формул логики высказываний, если предметные области $M_x = M_y = \{0,1\}$:

– $\forall xP(x,y)$;

– $\forall x(P(x,y) \vee Q)$;

– $\forall x\forall yP(x,y)$;

– $\forall x\exists yP(x,y)$;

– $\exists y\forall xP(x,y)$;

– $\exists x\exists yP(x,y)$.

31. Указать зоны действия кванторов, перечислить связанные и свободные предметные переменные формул:

– $\forall x(P(x) \vee Q(y)) = \forall xP(x) \vee Q(y)$;

– $\forall xP(x) \vee \forall xQ(x) = \forall x(P(x) \vee Q(x))$;

– $\exists x(P(x) \vee Q(x)) = \exists xP(x) \vee \forall xQ(x)$;

– $\forall x(P(x) \wedge Q(x)) = \forall xP(x) \wedge \forall xQ(x)$;

– $\exists x(P(x) \wedge Q(y)) = \exists xP(x) \wedge Q(y)$;

– $\exists x(P(x) \wedge Q(x)) = \exists xP(x) \wedge \exists xQ(x)$.

32. Доказать тождества, приведенные в задании 11.

33. Преобразовать формулы к почти нормальному виду и предваренной нормальной форме:

– $\overline{\exists}x(\forall yP(x,y,z) \to \forall uQ(x,u))$;

– $(\exists xP(x) \to \forall yQ(y)) \to R(z)$;

– $\neg(\exists x\forall yP(x,y) \wedge \exists x\forall yQ(x,y)$;

– $\exists x\forall y(\exists z(P(x,z) \wedge P(y,z)) \to \exists uR(x,y,u)$;

– $\exists x\forall yP(x,y) \vee \overline{\forall}x\exists yQ(x,y)$;

– $\neg(\forall x\forall y(\exists z(P(x,z) \wedge Q(y,z))) \to \exists uR(x,y,u)$;

– $\overline{\exists}uP(u) \to \neg(\forall y\forall uQ(y,u)) \to \forall xR(x)$;

– $\overline{\exists}x(\forall yP(x,y,z) \vee \forall uQ(x,u))$;

– $\overline{\forall}x(\exists yP(x,y,z) \to \exists uQ(x,u))$.

6 БУЛЕВЫ ФУНКЦИИ

6.1 Булево пространство

Упорядоченную совокупность булевых переменных ($x_1, x_2, \dots, x_n$) можно рассматривать как n-компонентный *булев вектор* $\boldsymbol{x}$. Число компонент вектора определяет его *длину*, или *размерность*. При фиксации значений всех переменных получается *набор значений* переменных, задаваемый булевым вектором длиной n, состоящим из констант 0 и 1. Совокупность различных n-компонентных булевых векторов $\boldsymbol{x}$ образует множество $E^n = E \times E \times \dots \times E$ (n-ю степень множества E), называемое *булевым пространством* размерностью n. Нетрудно подсчитать, что мощность $|E^n|$ булева пространства E^n равна 2^n.

Булево пространство размерностью n может также рассматриваться как множество всех подмножеств множества X. Например, булев вектор 1010 задает подмножество $\{x_1, x_3\}$ множества $\{x_1, x_2, x_3, x_4\}$.

Мерой булева пространства является расстояние между любыми его элементами. Расстояние между двумя элементами определяется *по Хэммингу* числом одноименных компонент пары сравниваемых элементов, имеющих несовпадающие значения. Чтобы найти расстояние по Хэммингу между булевыми векторами $\boldsymbol{x}_i$ и $\boldsymbol{x}_j$, необходимо взять покомпонентную дизъюнкцию с исключением (сумму по модулю два) этих векторов и подсчитать число единиц в ней. Это число и будет расстоянием по Хэммингу между булевыми векторами $\boldsymbol{x}_i$ и $\boldsymbol{x}_j$.

Например, расстояние по Хэммингу между булевыми векторами $\boldsymbol{x}_i = 1\,1\,0\,1$ и $\boldsymbol{x}_j = 1\,0\,1\,1$ равно 2, так как $1\,1\,0\,1 \oplus 1\,0\,1\,1 = 0\,1\,1\,0$.

Элементы, расстояние между которыми равно 1, называются *соседними*. Соседние элементы (и соответственно булевы векторы) отличаются ровно одной компонентой. Например, булевы векторы 1 1 0 0 и 1 0 0 0 являются соседними, а 1 1 0 0 и 1 0 1 0 – нет.

6.1.1 Графическое задание булева пространства

Булево пространство E^n можно представить в виде n-мерного куба – *гиперкуба*. 2^n вершин гиперкуба задают элементы булева пространства E^n, а $n\,2^{n-1}$ ребер связывают соседние элементы.

Графическое изображение гиперкуба получается методом удвоения размерности, начиная с 0-мерного куба. 0-мерный куб представляется одной вершиной, одномерный куб имеет две вершины, соответствующие элементам 0 и 1 булева пространства E^1 и соединенные ребром.

Двумерный куб имеет четыре вершины (в два раза больше, чем одномерный), соответствующие элементам 0 0, 1 0, 1 1 и 0 1 булева пространства E^2, и четыре ребра, соединяющие вершины, представляющие соседние элементы (рисунок 6.1). В общем случае изображение $(n+1)$-мерного куба получается из изображения n-мерного куба смещением его в некотором направлении, условно ортогональном ранее использованным, и включением в новое изображение как исходного, так и смещенного изображения вместе с трассами вершин, проходимыми при смещении.

Примеры такого построения до значения $n = 5$ включительно показаны на рисунках 6.1 и 6.2. Там же показано, какие конкретно элементы булева пространства представлены различными вершинами.

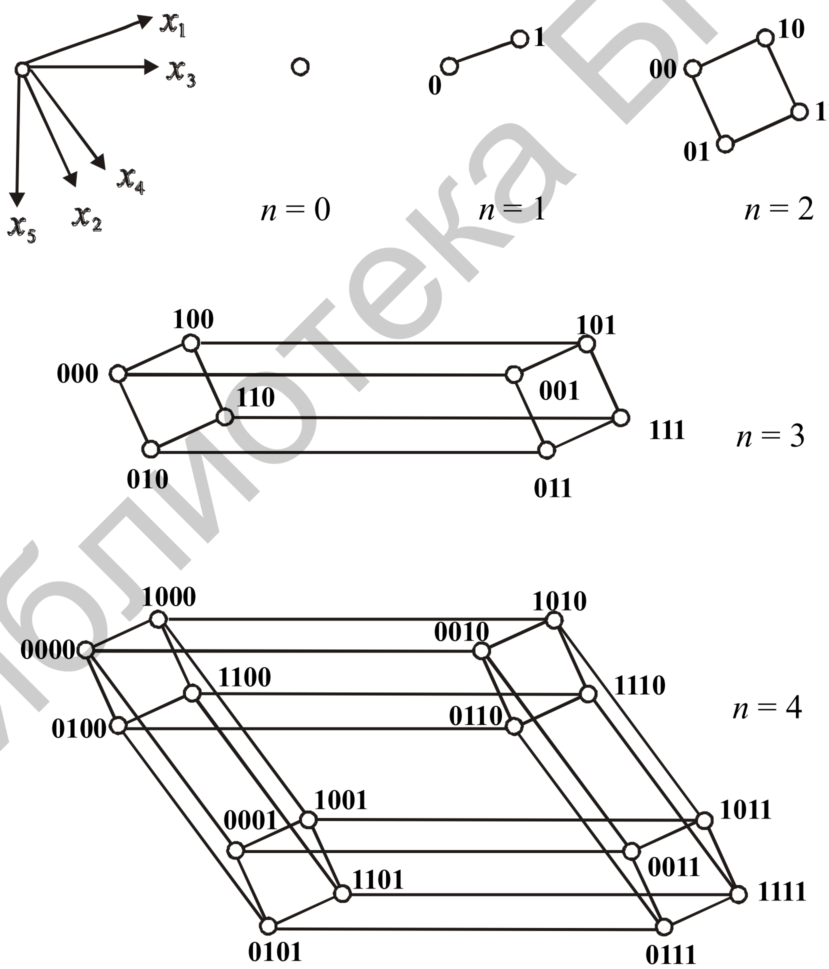


Рисунок 6.1 – Построение четырехмерного куба

В левом верхнем углу рисунка 6.1 показаны направления движения при переходе от одной вершины гиперкуба к другой, соседней с ней, и наименования компонент булева вектора $\boldsymbol{x} = (x_1, x_2, x_3, x_4, x_5)$, изменяющих свое значение с 0 на 1.

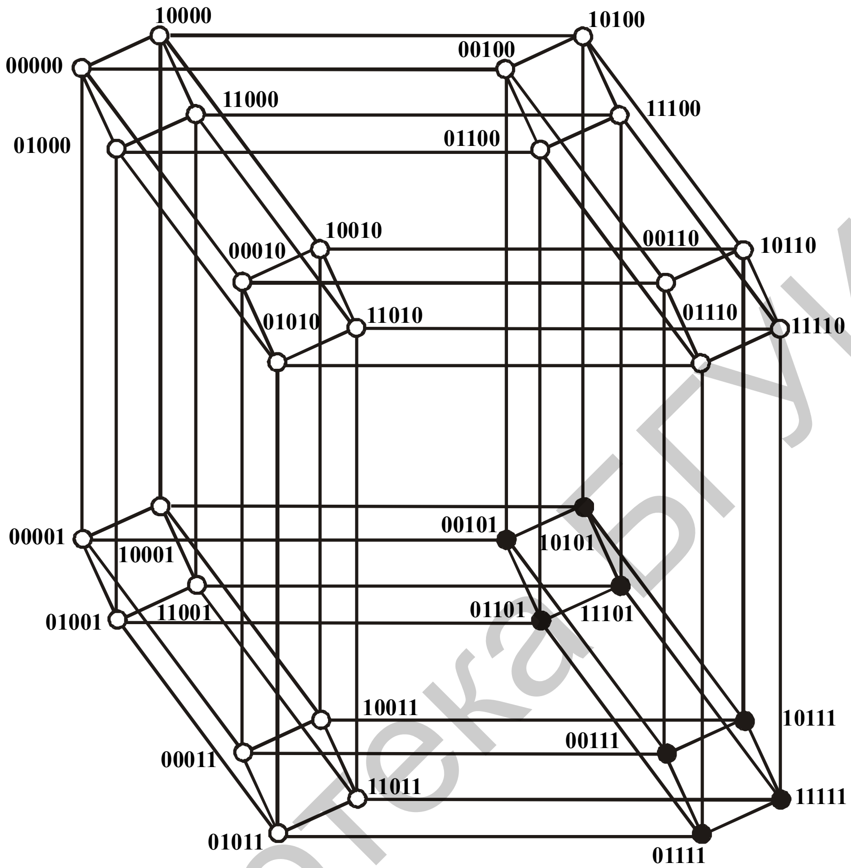


Рисунок 6.2 – Пятимерный куб с заданным на нем трехмерным подкубом

В n-мерном кубе (для любого n) можно выделять подкубы с меньшей размерностью. Наименьшим подкубом является любая вершина – 0-мерный подкуб n-мерного куба. Одномерный подкуб – пара вершин, связанных ребром, трехмерный подкуб – куб, состоящий из восьми (2^3) вершин. Наибольшим подкубом является сам n-мерный куб.

Вершина n-мерного куба представляет n-мерный булев вектор – элемент булева пространства E^n. Подкуб, в отличие от вершины, задает подмножество булевых векторов – элементов пространства E^n. Его удобно задавать n-мерным троичным вектором, в котором i-я компонента имеет определенное значение (0 или 1), если такое значение имеют i-компоненты векторов всех вершин подкуба, в противном случае i-й компоненте присваивается неопределенное значение «–», трактуемое как «0 или 1».

Например, трехмерный подкуб, состоящий из зачерненных вершин пятимерного куба на рисунке 6.2, задается троичным вектором – – 1 – 1.

6.1.2 Интервалы булева пространства

Троичный вектор $\boldsymbol{u}$ интерпретируется как множество $I(\boldsymbol{u})$ всех таких булевых векторов, которые получаются из него всевозможными подстановками значений 0 и 1 вместо неопределенного значения «–». Если в троичном векторе имеется k неопределенных значений «–», то он порождает таким образом 2^k булевых вектора. Например, троичный вектор $\boldsymbol{u}$ = 0 – 1 – 1 порождает множество $I(\boldsymbol{u})$ = {0 0 1 0 1, 0 0 1 1 1, 0 1 1 0 1, 0 1 1 1 1}.

Считая, что 1 > 0, можно определить следующие *отношения* между булевыми векторами $\boldsymbol{x}$ и $\boldsymbol{y}$ одной размерности:

– равенства: $\boldsymbol{x} = \boldsymbol{y}$, если их одноименные компоненты равны, т. е. $x^i = y^i$ для всех i;

– не меньше (не больше): $\boldsymbol{x} \geq \boldsymbol{y}$, если $x^i \geq y^i$ для всех i;

– больше (меньше): $\boldsymbol{x} > \boldsymbol{y}$, если $x^i \geq y^i$ для всех i и существует такое j, что $x^j > y^j$.

Например, булев вектор 0 1 1 больше векторов 0 0 0, 0 0 1, 0 1 0, но меньше вектора 1 1 1 и несравним с векторами 1 0 0, 1 0 1 и 1 1 0.

Для множества $I(\boldsymbol{u})$ булевых векторов, порождаемых троичным вектором $\boldsymbol{u}$, отношения «<» и «>» («≤» и «≥») являются отношениями частичного порядка: они антисимметричны и транзитивны, а первые два и иррефлексивны, но свойство дихотомии для этих отношений не имеет места, так как не каждая пара булевых векторов сравнима между собой. Частично упорядоченное отношением «>» (или «<») множество $I(\boldsymbol{u})$ имеет минимальный и максимальный элементы. Элемент $\boldsymbol{x}_p \in I(\boldsymbol{u})$ называется минимальным, если не существует элемента $\boldsymbol{x}_i \in I(\boldsymbol{u})$, такого, что $\boldsymbol{x}_p > \boldsymbol{x}_i$. Элемент $\boldsymbol{x}_p \in I(\boldsymbol{u})$ называется максимальным, если не существует элемента $\boldsymbol{x}_i \in I(\boldsymbol{u})$, такого, что $\boldsymbol{x}_i > \boldsymbol{x}_p$. Все остальные булевы векторы из $I(\boldsymbol{u})$ находятся между минимальным и максимальным элементами, т. е. каждый из них больше минимального и меньше максимального.

Таким образом, множество $I(\boldsymbol{u})$ булевых векторов, представимое троичным вектором $\boldsymbol{u}$, образует в булевом пространстве E^n *интервал*, имеющий минимальный и максимальный элементы. Минимальный элемент получается подстановкой нулей вместо всех значений «–» троичного вектора, а максимальный – подстановкой единиц.

Рангом интервала называется число компонент его векторного представления, имеющих значение 1 или 0. Интервал ранга m булева пространства E^n

состоит из 2^{n-m} элементов. Он задается $(n-m)$-мерным подкубом на n-мерном гиперкубе.

Например, троичный вектор $- - 0\,0\,1$ при такой интерпретации рассматривается как интервал ранга 3 в пятимерном булевом пространстве, который состоит из четырех элементов $0\,0\,0\,0\,1$, $0\,1\,0\,0\,1$, $1\,0\,0\,0\,1$ и $1\,1\,0\,0\,1$, причем $0\,0\,0\,0\,1$ – минимальный, а $1\,1\,0\,0\,1$ – максимальный элементы этого интервала.

Троичный вектор ***u*** задает также некоторую элементарную конъюнкцию, которая легко находится, если принять, что значениями 0 и 1 в векторе отмечены переменные, входящие в конъюнкцию соответственно со знаком отрицания и без него, а значением «–» отмечены переменные, отсутствующие в конъюнкции. Например, троичный вектор $-0\,1-1$ задает конъюнкцию $\bar{x}_2 x_3 x_5$. Элементарная конъюнкция i-го ранга (содержащая i букв) представляется интервалом i-го ранга.

Далее будем рассматривать векторы одинаковой размерности, полагая, что булевы векторы представляют собой частный случай троичных векторов. Рассмотрим отношения, в которых могут находиться троичные векторы ***u*** и ***v***.

1. Отношение *равенства* векторов ***u*** = ***v*** определяется стандартным образом как покомпонентное равенство.

2. *Ортогональность*. Троичные векторы ***u*** и ***v*** ортогональны по i-й компоненте, если и только если i-я компонента принимает значение 0 в одном из векторов и значение 1 – в другом. Векторы ***u*** и ***v*** *ортогональны* (обозначается как ***u*** ort ***v***), если они ортогональны по крайней мере по одной из компонент.

3. *Пересечение*. Неортогональные векторы ***u*** и ***v*** находятся в отношении *пересечения* (***u*** ins ***v***).

Например,

$$0-1\,1---0\,0 \;\text{ ort }\; 0\,1\,0\,1-0-1\,0,$$
$$0-1\,1---0\,0 \;\text{ ins }\; 0\,1\,1--0--0,$$
$$0-1\,1---0\,0 \;\text{ ort }\; 0-0\,1-0-0\,0,$$
$$0-1\,1---0\,0 \;\text{ ins }\; 0\,0\,1\,1\,0\,1\,0\,0\,0.$$

4. *Поглощение*. Вектор ***u*** *поглощает* вектор ***v*** (***u*** abs ***v***), если и только если все компоненты вектора ***u***, значения которых отличны от «–», совпадают с соответствующими компонентами вектора ***v***. Можно заметить, что если компонента поглощаемого вектора имеет значение «–», то соответствующая компонента поглощающего вектора также имеет значение «–». Например,

$$0-1\,1---0\,0 \;\text{ abs }\; 0\,1\,1\,1-0-0\,0.$$

В общем случае троичный вектор $\boldsymbol{u}$ поглощает все векторы (и только их), которые образуются из него подстановкой на места символов «–» всевозможных комбинаций из 0, 1 и «–». Таким образом, если в троичном векторе имеется k значений «–», то он поглощает 3^k различных вектора (булевых и троичных). Например, вектор 0 1 – – поглощает векторы 0 1 – –, 0 1 – 0, 0 1 – 1, 0 1 0 –, 0 1 1 –, 0 1 0 0, 0 1 0 1, 0 1 1 0, 0 1 1 1.

5. *Смежность*. Векторы $\boldsymbol{u}$ и $\boldsymbol{v}$ находятся в отношении *смежности* ($\boldsymbol{u}$ adj $\boldsymbol{v}$), если и только если они ортогональны и только по одной из компонент. В случаях если есть необходимость в уточнении этого отношения, говорят, что векторы $\boldsymbol{u}$ и $\boldsymbol{v}$ смежны по i-й компоненте ($\boldsymbol{u}$ adj_i $\boldsymbol{v}$).

6. *Соседство*. Если векторы $\boldsymbol{u}$ и $\boldsymbol{v}$ смежны по некоторой i-й компоненте и, кроме того, равны по значениям всех остальных компонент, то они находятся в отношении *соседства* ($\boldsymbol{u}$ nei $\boldsymbol{v}$) или соседства по i-й компоненте ($\boldsymbol{u}$ nei_i $\boldsymbol{v}$). Например,

$$0 - 1\,1 - - - 0\,0 \ \ \mathrm{adj} \ \ 0\,1\,1\,1 - 0 - 1\,0;$$
$$0 - 1\,1 - - - 0\,0 \ \ \mathrm{adj}_3 \ \ 0 - 0\,1\,0\,1\,1\,0\,0;$$
$$0 - 1\,1 - - - 0\,0 \ \ \mathrm{nei} \ \ 0 - 1\,1 - - - 0\,1;$$
$$0 - 1\,1 - - - 0\,0 \ \ \mathrm{nei}_4 \ \ 0 - 1\,0 - - - 0\,0.$$

Нетрудно заметить, что отношения ортогональности, пересечения, смежности и соседства являются симметричными, т. е. из $\boldsymbol{u}$ ort $\boldsymbol{v}$ всегда следует $\boldsymbol{v}$ ort $\boldsymbol{u}$ и т. д. Отношение поглощения несимметрично, но обладает свойством транзитивности, т. е. из $\boldsymbol{u}$ abs $\boldsymbol{v}$ и $\boldsymbol{v}$ abs $\boldsymbol{w}$ всегда следует $\boldsymbol{u}$ abs $\boldsymbol{w}$.

Используя понятие интервала, можно дать следующую интерпретацию отношения ортогональности между троичными векторами: если $\boldsymbol{u}$ ort $\boldsymbol{v}$, то соответствующие им интервалы $I(\boldsymbol{u})$ и $I(\boldsymbol{v})$ булева пространства E^n находятся в отношении непересечения: $I(\boldsymbol{u}) \cap I(\boldsymbol{v}) = \varnothing$. И наоборот, если $I(\boldsymbol{u}) \cap I(\boldsymbol{v}) = \varnothing$, то $\boldsymbol{u}$ ort $\boldsymbol{v}$. В символической записи это утверждение выглядит следующим образом:

$$\boldsymbol{u} \ \mathrm{ort} \ \boldsymbol{v} \Leftrightarrow I(\boldsymbol{u}) \cap I(\boldsymbol{v}) = \varnothing.$$

Аналогичную интерпретацию имеют отношения пересечения и поглощения:

$$\boldsymbol{u} \ \mathrm{ins} \ \boldsymbol{v} \Leftrightarrow I(\boldsymbol{u}) \cap I(\boldsymbol{v}) \neq \varnothing;$$
$$\boldsymbol{u} \ \mathrm{abs} \ \boldsymbol{v} \Leftrightarrow I(\boldsymbol{u}) \supseteq I(\boldsymbol{v}).$$

Например:

$$0{-}11{-}00 \text{ ort } 0101{-}01 \Leftrightarrow$$
$$\{0011000, 0011100, 0111000, 0111100\} \cap \{0101001, 0101101\} = \varnothing;$$
$$0{-}11{-}00 \text{ ins } 011{-}0{-}0 \Leftrightarrow \{0011000, 0011100, 0111000, 0111100\} \cap$$
$$\cap \{0110000, 0110010, 0111000, 0111010\} = \{0111000\} \neq \varnothing;$$
$$0{-}11{-}00 \text{ abs } 0111{-}00 \Leftrightarrow \{0011000, 0011100, 0111000, 0111100\} \supseteq$$
$$\supseteq \{0111000, 0111100\}.$$

Аналогично отношения смежности и соседства между троичными векторами интерпретируются как соответствующие отношения между интервалами.

6.1.3 Развертка гиперкуба на плоскость

Если удалить два ребра трехмерного куба, то полученную двумерную фигуру можно разместить на плоскости. Если удалить еще три ребра, то получим одномерную фигуру (рисунок 6.3).

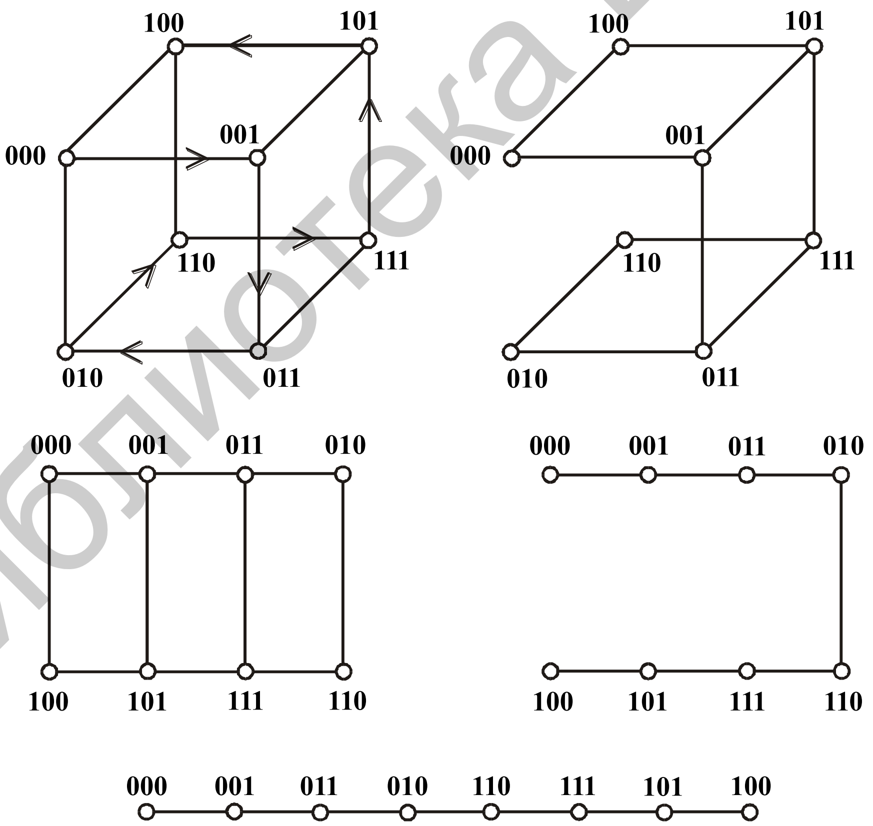


Рисунок 6.3 – Получение двумерной и одномерной разверток трехмерного куба

Последовательность булевых векторов, которыми оказываются пронумерованы вершины одномерной развертки куба, представляет *код Грея* для натуральных чисел, известный также под названием *симметричного кода*, его соседние натуральные числа кодируются соседними булевыми векторами.

Алгоритм получения кода Грея заключается в итеративном повторении одной и той же процедуры изменения последней полученной кодовой комбинации. Каждая следующая комбинация кода Грея получается из предыдущей путем изменения значения одной компоненты. При этом всегда делается попытка изменить значение самой правой компоненты. Если результатом является полученная ранее комбинация, то данная операция отвергается и делается попытка изменить значение следующей слева компоненты и т. д.

Симметричность кода Грея длиной k проявляется в следующем. Множество последних 2^{k-1} кодовых комбинаций получается симметричным отображением первых 2^{k-1} кодовых комбинаций относительно оси, делящей все множество кодовых комбинаций пополам, и изменением значения крайней слева компоненты всех отображенных кодов с 0 на 1.

Для примера покажем получение четырехкомпонентного кода Грея:

1. 0 0 0 0	9. 1 1 0 0
2. 0 0 0 1	10. 1 1 0 1
3. 0 0 1 1	11. 1 1 1 1
4. 0 0 1 0	12. 1 1 1 0
5. 0 1 1 0	13. 1 0 1 0
6. 0 1 1 1	14. 1 0 1 1
7. 0 1 0 1	15. 1 0 0 1
8. 0 1 0 0	16. 1 0 0 0

В примере приведены оси симметрии соответственно (сверху вниз) для однокомпонентного, двухкомпонентного, трехкомпонентного и четырехкомпонентного кодов Грея.

На рисунке 6.4 продемонстрирован способ получения двумерной развертки пятимерного гиперкуба (см. рисунок 6.2).

Пятимерный куб последовательно разворачивается путем удаления некоторых ребер и соответствующего «выпрямления» в пространство меньшей размерности. В результате получается двумерная матричная фигура (рисунок 6.5, *а*). Ее узлам соответствуют элементы булева пространства E^5, получаемые соединением вектора, отмечающего горизонтальную линию, с вектором, отмечающим вертикальную линию. При этом левый верхний узел представляет набор 0 0 0 0 0, соседний с ним справа – набор 0 0 1 0 0, соседний с 0 0 0 0 0 снизу – набор 0 1 0 0 0 и т. д. Последовательность булевых векторов, которыми оказываются пронумерованы как горизонтальные, так и вертикальные линии, представляет код Грея. Заметим, что порядок следования компонент в кодах вертикальных линий инвертирован – младший разряд оказывается слева.

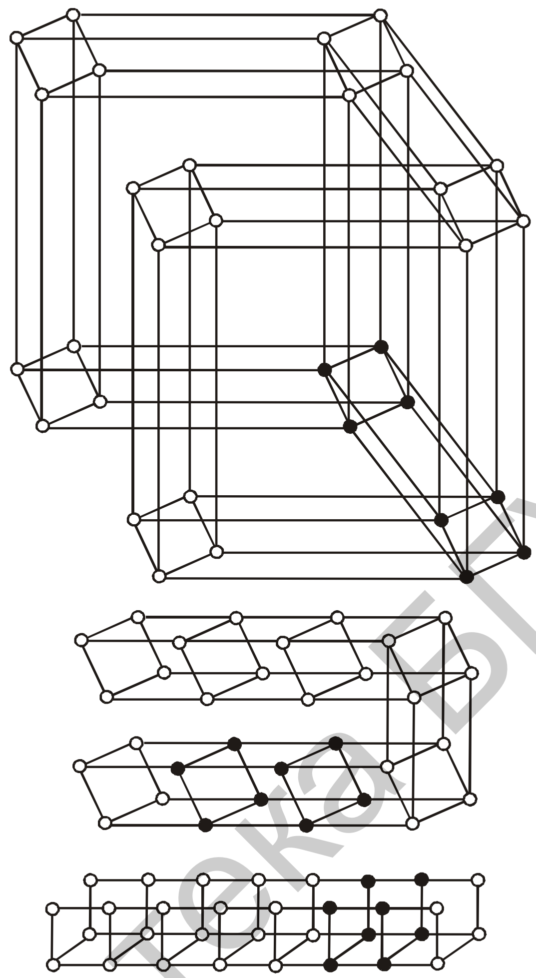
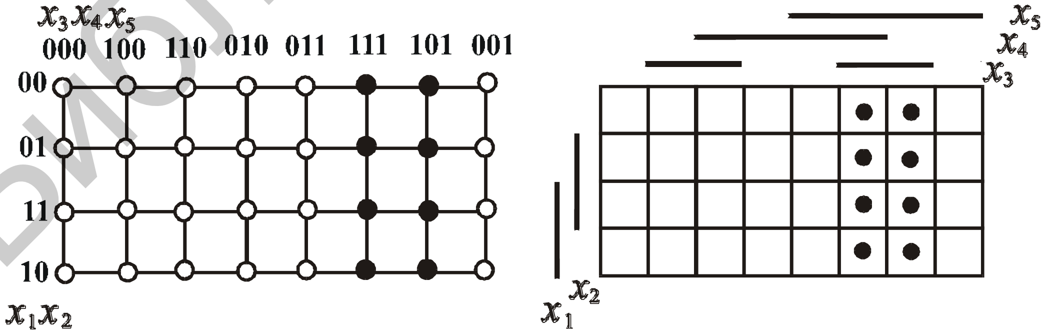


Рисунок 6.4 – Получение двумерной развертки пятимерного гиперкуба



а *б*

а – двумерная матричная структура; *б* – двумерная таблица

Рисунок 6.5 – Развертка пятимерного куба

Модифицируя полученное изображение, можно перейти к более удобной для последующего использования форме (рисунок 6.5, *б*). Здесь узлы заменены клетками двумерной таблицы, а показанные сверху и слева пронумерованные полужирные линии (иногда представленные несколькими отрезками) соответствуют булевым переменным, образующим пространство E^5, и проведены напротив тех клеток, которые представляют наборы со значением 1 соответствующих компонент.

6.1.4 Карта Карно

Двумерная таблица (см. рисунки 6.5, *б* и 6.6), которая является разверткой гиперкуба на плоскость, называется *картой Карно* (или диаграммой *Вейча*). Карту Карно, представляющую развертку *n*-мерного гиперкуба, будем называть далее *n*-мерной. На рисунке 6.6 в каждой ее клетке показан соответствующий элемент булева пространства E^n.

Аналогично тому, как *n*-компонентный код Грея получается симметричным отображением кодовых комбинаций (*n*–1)-компонентного кода Грея (с добавлением слева новой компоненты и сменой ее значения с 0 на 1 в кодах соответственно старого и нового множества), *n*-мерная карта Карно может быть получена из (*n*-1)-мерной ее симметричным отображением. На рисунке 6.7 показан процесс последовательного наращивания размерности карты Карно: от одномерной до шестимерной. Стрелкой показано направление симметричного отображения относительно выделенной оси симметрии.

x_5 x_4 x_3

00000	**00100**	**00110**	**00010**	**00011**	**00111**	**00101**	**00001**
01000	**01100**	**01110**	**01010**	**01011**	**01111**	**01101**	**01001**
11000	**11100**	**11110**	**11010**	**11011**	**11111**	**11101**	**11001**
10000	**10100**	**10110**	**10010**	**10011**	**10111**	**10101**	**10001**

x_2 x_1

Рисунок 6.6 – Пятимерная карта Карно

Следует заметить, что *n*-мерная карта Карно может быть и не квадратной, а сильнее вытянутой по вертикали или по горизонтали (в частности, это неизбежно, если *n* – нечетное). Полужирной линии карты Карно может быть поставлена в соответствие любая булева переменная, т. е. множество всех переменных разбивается на два подмножества (соответствующих горизонтальным и вертикальным полужирным линиям) произвольным образом, произвольным же образом устанавливается порядок их следования в этих подмножествах. К примеру, карте Карно на рисунке 6.7 (n = 6) соответствует разбиение переменных $\{\{x_6 x_2 x_1\}, \{x_2 x_3 x_5\}\}$, а карте на рисунке 6.8 – разбиение $\{\{x_1 x_2 x_3\}, \{x_4 x_5 x_6\}\}$.

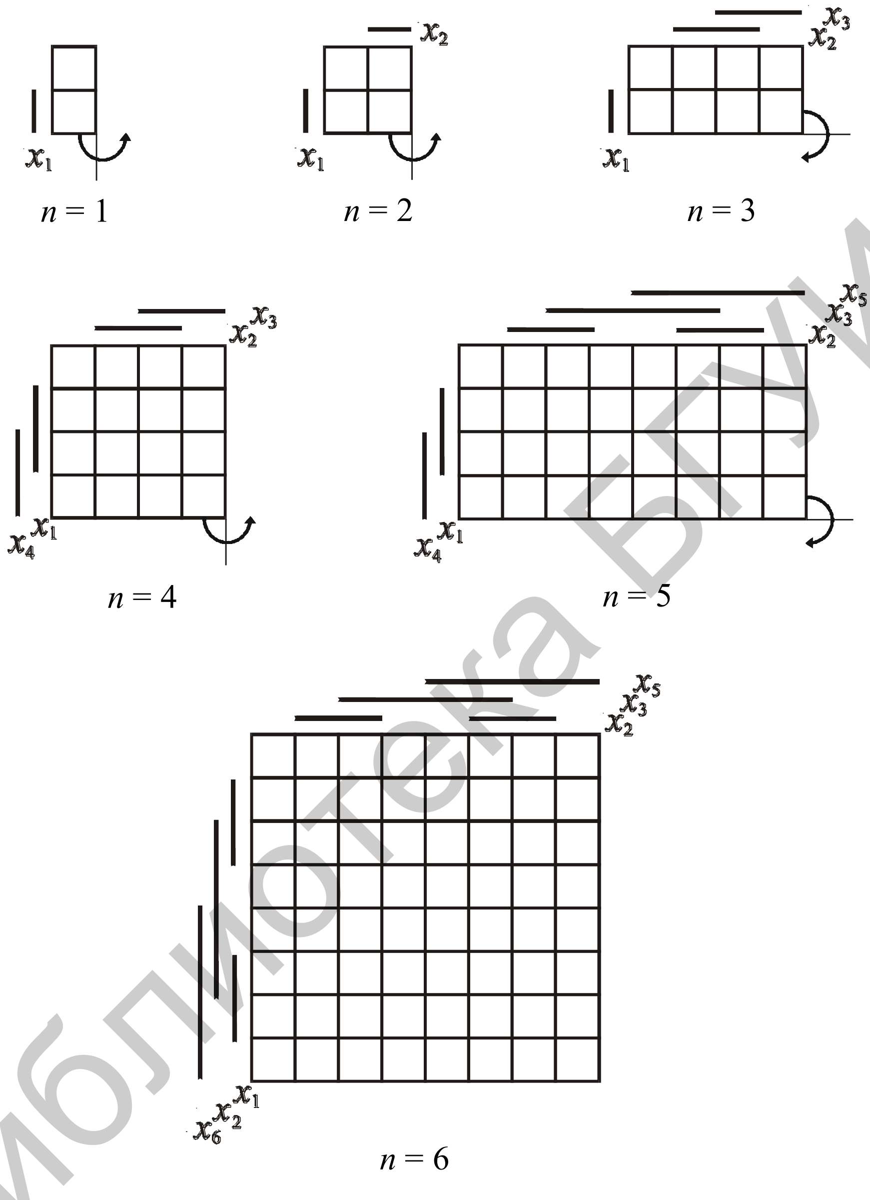


Рисунок 6.7 – Получение шестимерной карты Карно

Одно из достоинств карты Карно заключается в том, что соседние клетки ее изображения всегда представляют соседние элементы отображаемого булева пространства, т. е. образуют двухэлементный интервал. Например, соседние клетки карты Карно, с элементами 0 0 0 0 0 и 0 1 0 0 0 (см. рисунок 6.6), порождают двухэлементный интервал, представляемый троичным вектором 0 – 0 0 0. Интервалом оказывается и квадрат из четырех расположенных по соседству клеток изображения. Например, соседние клетки с элементами 0 0 0 0 0, 0 1 0 0 0, 0 0 0 0 1, 0 1 0 0 1 (см. рисунок 6.6) порождают интервал, представляемый троичным вектором 0 – 0 0 – .

В общем случае соседние элементы булева пространства представляются элементами карты Карно, которые симметричны относительно некоторой ее оси (горизонтальной или вертикальной) и находятся в зоне ее симметрии. Оси симметрии разделяют элементы карты Карно, соседние по одной переменной. Эти элементы образуют *зону симметрии* оси. Например, первая сверху горизонтальная ось симметрии, показанная на рисунке 6.8, имеет ранг 1 и разделяет пары соседних элементов, принадлежащих первой и второй строке карты. Эти пары элементов находятся непосредственно выше и ниже от оси и являются соседними по переменной x_3:

0 0 0 0 0 0 nei 0 0 1 0 0 0; 0 0 0 1 0 0 nei 0 0 1 1 0 0; 0 0 0 1 1 0 nei 0 0 1 1 1 0 и т. д.

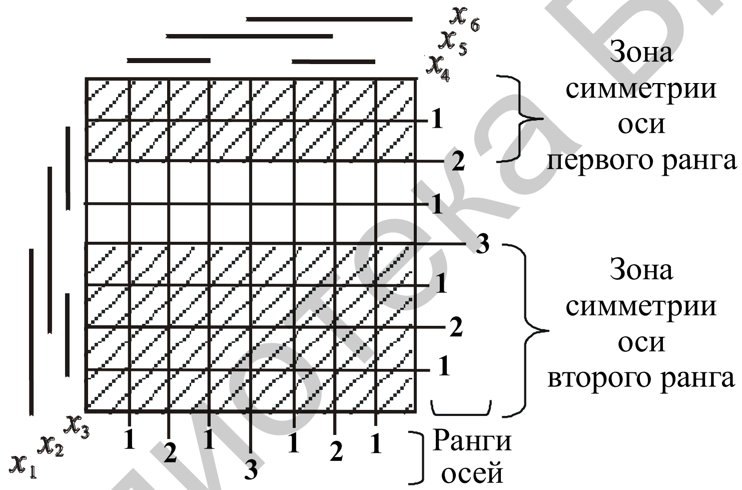


Рисунок 6.8 – Зоны симметрии шестимерной карты Карно

Вторая сверху горизонтальная ось симметрии имеет ранг 2 и разделяет пары элементов строк, находящихся на одном и том же расстоянии (1 или 2) от оси и являющихся соседними по переменной x_2. Например, соседними являются следующие пары элементов, находящихся на расстоянии 1 и 2 от оси:

0 0 1 0 0 0 nei 0 1 1 0 0 0; 0 0 1 1 0 0 nei 0 1 1 1 0 0; 0 0 1 1 1 0 nei 0 1 1 1 1 0 и т. д.

0 0 0 0 0 0 nei 0 1 0 0 0 0; 0 0 0 1 0 0 nei 0 1 0 1 0 0; 0 0 0 1 1 0 nei 0 1 0 1 1 0 и т. д.

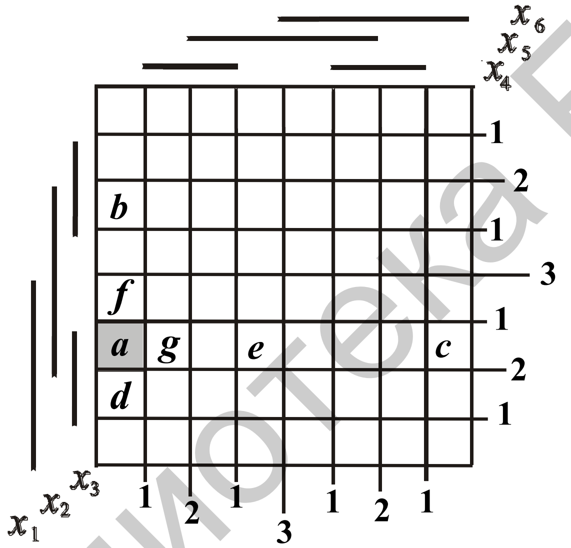
В общем случае зона симметрии оси k-го ранга образуется элементами, расположенными от оси не далее чем на расстоянии 2^{k-1}. Как определить ранги осей, показано на рисунке 6.8, где изображена шестимерная карта Карно. Там же приведены примеры зон симметрии вертикальных и горизонтальных осей.

На рисунке 6.9 приведена шестимерная карта Карно. На ней выделены клетка, представляющая элемент ***a*** = 1 1 1 0 0 0 булева пространства E^6, и клетки, симметричные ей относительно всех осей и представляющие все соседние с ***a*** элементы. Компоненты, по которым наборы являются соседними с ***a***, выделены на рисунке полужирным шрифтом. Элемент ***a*** = 1 1 1 0 0 0 является соседним с тремя элементами относительно горизонтальных осей симметрии ранга 3, 2 и 1, соответствующих переменным x_1, x_2 и x_3:

1 1 1 0 0 0 nei **0** 1 1 0 0 0; 1 1 1 0 0 0 nei 1 **0** 1 0 0 0; 1 1 1 0 0 0 nei 1 1 **0** 0 0 0,

и с тремя элементами относительно вертикальных осей симметрии ранга 3, 2 и 1, соответствующих переменным x_6, x_5 и x_4:

1 1 1 0 0 0 nei 1 1 1 0 0 **1**; 1 1 1 0 0 0 nei 1 1 1 0 **1** 0; 1 1 1 0 0 0 nei 1 1 1 **1** 0 0.



a = 111000

b = **0**11000

c = 11100**1**

d = 1**0**1000

e = 1110**1**0

f = 11**0**000

g = 111**1**00

Рисунок 6.9 – Элементы, симметричные выделенному элементу относительно всех осей симметрии шестимерной карты Карно

На карте Карно легко выявляются интервалы булева пространства. При их распознавании полезно помнить, что любая клетка n-мерной карты Карно представляет собой интервал n-го ранга и что два интервала i-го ранга, изображения которых симметричны относительно некоторой оси и полностью находятся в зоне ее симметрии, образуют интервал $(i–1)$-го ранга. Напомним, что интервал n-го ранга задает полную элементарную конъюнкцию, а интервал i-го ранга – элементарную конъюнкцию i-го ранга (содержащую i букв).

Например, выделенный на рисунке 6.9 элемент ***a*** = 1 1 1 0 0 0 карты Карно представляет интервал шестого ранга, задающий элементарную конъюнкцию $x_1 x_2 x_3 \bar{x}_4 \bar{x}_5 \bar{x}_6$. Этот элемент вместе с любым другим симметричным ему относительно какой-либо оси симметрии элементом образует интервал пятого ранга. Например, симметричные относительно оси второго ранга по горизонтали элементы, представляющие $\boldsymbol{a}$ = 1 1 1 0 0 0 и $\boldsymbol{e}$ = 1 1 1 0 1 0, образуют интервал пятого ранга 1 1 1 0 – 0.

Интервалы первого ранга, задающие одноэлементные элементарные конъюнкции, называются *базовыми*. Они представляют собой совокупности клеток, расположенных в строках или столбцах карты Карно, отмеченных соответствующей полужирной линией либо ее дополнением. Любая другая элементарная конъюнкция задается интервалом, образуемым пересечением всех базовых интервалов, соответствующих входящим в конъюнкцию литералам.

6.2 Булевы функции

6.2.1 Основные определения

Пусть $x_1, x_2, ..., x_n$ – некоторые булевы переменные, т. е. переменные, принимающие значение из двухэлементного множества $E = \{0, 1\}$. Упорядоченную совокупность булевых переменных $(x_1, x_2, \ldots, x_n)$ можно рассматривать как n-компонентный булев вектор $\boldsymbol{x}$. Функция n аргументов, обозначаемая как $y = f(\boldsymbol{x}) = f(x_1, x_2, ..., x_n)$, называется *булевой*, если ее аргументы и сама функция есть булевы переменные. Булева функция задает отображение n-мерного булева пространства E^n в двухэлементное множество E:

$$f(\boldsymbol{x}):\ E^n \to E.$$

Так как булево пространство E^n, содержащее 2^n элементов, отображается в двухэлементное множество E, то число различных булевых функций n аргументов равно числу различных булевых векторов длиной 2^n, т. е. 2^{2^n} .

Областью определения булевой функции $f(\boldsymbol{x}) = f(x_1, x_2, ..., x_n)$ является совокупность тех наборов значений вектора $\boldsymbol{x}$, для которых определены ее значения. Если значения булевой функции определены для всех 2^n наборов значений вектора $\boldsymbol{x}$, то она называется *полностью определенной*, в противном случае – *не полностью определенной* или *частичной*. Далее ограничимся, если не оговаривается иное, рассмотрением только полностью определенных булевых функций.

Задание полностью определенной булевой функции разделяет булево пространство E^n, обозначаемое далее через M, на два множества: M_f^1 и M_f^0. Совокупность элементов булева пространства M, на которых функция $y = f(\boldsymbol{x})$ принимает значение 1, называется множеством M_f^1 ее *единичных* значений или *характеристическим множеством* функции. Соответственно, совокупность наборов значений переменных, на которых функция обращается в 0, – множеством M_f^0 ее *нулевых* значений.

Из определения полностью определенной булевой функции следует, что для нее выполняются следующие условия:

$$M_f^1 \cap M_f^0 = \varnothing, \quad M_f^1 \cup M_f^0 = M.$$

Из этих условий вытекает, что, для того чтобы задать булеву функцию, достаточно перечислить элементы одного из множеств M_f^1 и M_f^0, обычно ее характеристического множества M_f^1. Две булевы функции $f(\boldsymbol{x})$ и $g(\boldsymbol{x})$ *равны*, если и только если их характеристические множества совпадают, т. е. $M_f^1 = M_g^1$.

Значения частичной булевой функции определены (принимают значения 0 и 1) не для всех наборов значений аргументов, т. е.

$$M_f^1 \cup M_f^0 \subset M.$$

Частичная булева функция $f(\boldsymbol{x})$ задает разбиение булева пространства M уже не на два, а на три подмножества:

– M_f^1, на котором функция принимает значение 1;

– M_f^0, на котором функция принимает значение 0;

– M_f^-, на котором функция не определена.

Множество $M_f^1 \cup M_f^0$ элементов булева пространства M называется областью определения функции $f(\boldsymbol{x})$. Множество M_f^- называется областью неопределенных (или безразличных) значений функции $f(\boldsymbol{x})$. Неопределенное значение функции также обозначается символом «–».

Из определения частичной булевой функции следует, что для нее выполняются следующие условия:

$$M_f^1 \cap M_f^0 = \varnothing, \quad M_f^1 \cap M_f^- = \varnothing, M_f^0 \cap M_f^- = \varnothing, M_f^1 \cup M_f^0 \cup M_f^- = M.$$

Следовательно, чтобы задать частичную булеву функцию, достаточно задать лишь какие-либо два из множеств M_f^1, M_f^0, M_f^-, например, выбрав те из них, которые содержат меньше элементов. Обычно задается одна из пар M_f^1, M_f^0 или M_f^1, M_f^-, включающих характеристическое множество M_f^1 функции.

Частичная булева функция задает отображение булева пространства E^n (состоящего из 2^n элементов) в трехэлементное множество:

$$f\colon E^n \to \{1, 0, -\}.$$

Соответственно число различных частичных булевых функций n аргументов равно числу различных булевых векторов длины 3^n, т. е. 2^{3^n}, что значительно превышает число различных полностью определенных булевых функций (2^{2^n}).

Две частичные булевы функции $f(\boldsymbol{x})$ и $g(\boldsymbol{x})$ *равны*, если и только если совпадают пары задающих их множеств, т. е.

$$f(\boldsymbol{x}) = g(\boldsymbol{x}) \Leftrightarrow (M_f^1 = M_g^1) \wedge (M_f^0 = M_g^0)$$

или

$$f(\boldsymbol{x}) = g(\boldsymbol{x}) \Leftrightarrow (M_f^1 = M_g^1) \wedge (M_f^- = M_g^-).$$

Важным отношением между частичными функциями (а также между ними и полностью определенными функциями) является *отношение реализации*. Рассматривая две булевы функции $f(\boldsymbol{x})$ и $g(\boldsymbol{x})$ (определенные на одном булевом пространстве M), говорят, что функция $g(\boldsymbol{x})$ *реализует* функцию $f(\boldsymbol{x})$, если значение функции $g(\boldsymbol{x})$ совпадает со значением функции $f(\boldsymbol{x})$ на всей области определения последней, т. е. если

$$M_f^1 \subseteq M_g^1 \text{ и } M_f^0 \subseteq M_g^0$$

(вне области $M_f^1 \cup M_f^0$ значения функции $g(\boldsymbol{x})$ произвольны).

6.2.2 Способы представления булевых функций

1. *Теоретико-множественное задание*. Булева функция задается множествами M_f^1 и M_f^0 (или одним из них), в которых перечисляются наборы значений аргументов, на которых она принимает значения 1 и 0.

2. *Табличное задание*. Простейшим, хотя и громоздким, способом задания булевой функции является табличное (*таблица истинности*). Таблица состоит из двух частей (см. таблицу 6.1): в ее левой части перечисляются все возможные наборы значений аргументов (2^n – для функции n аргументов), а в правой части – значения функции на этих наборах. Обычно используется стандартный порядок перечисления наборов, интерпретируемых в данном случае как позиционные двоичные коды последовательности натуральных чисел (0, 1, 2, 3, 4, …) с основанием 2. Этот порядок называется *лексикографическим* и подобен порядку, принятому в словарях.

Например, таблица 6.1 задает булеву функцию $f(x_1, x_2, x_3)$, принимающую значение 0 на наборах 0 0 0, 1 0 0 и 1 1 0 и значение 1 – на остальных наборах.

Таблица 6.1 – Табличное задание булевой
функции $f(x_1, x_2, x_3)$

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

3. *Матричное задание*. Табличное представление функции $f(\boldsymbol{x})$ можно сжать, ограничившись перечислением элементов ее характеристического множества M_f^1. Множество M_f^1 удобно отображать матрицей, в которой столбцы поставлены в соответствие с аргументами функции, а строки представляют наборы множества M_f^1. Элементами такой матрицы могут служить лишь 0 и 1, т. е. матрица оказывается булевой. Например, функция, заданная таблицей 6.1, представляется следующей булевой матрицей:

$$\begin{array}{ccc} x_1 & x_2 & x_3 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{array}$$

Представляемое матрицей характеристическое множество M_f^1 рассматривается как неупорядоченное множество без повторений. Назовем две булевы матрицы эквивалентными, если множества их строк равны, т. е. для любой строки каждой из этих матриц найдется равная ей строка в другой матрице. Очевидно, что эквивалентные булевы матрицы могут различаться только порядком строк и задают равные функции.

Компактность представления характеристического множества M_f^1 можно повысить, если использовать троичные векторы. В этом случае значения булевой функции будут задаваться не на отдельных элементах булева пространства E^n переменных $x_1, x_2, \ldots, x_n$, а на его интервалах.

Введем в рассмотрение *троичные* матрицы, строками которых служат троичные векторы. Положим, что троичная матрица эквивалентна булевой матрице, получаемой из нее заменой каждой троичной строки на порождаемую ею совокупность булевых строк и последующим «приведением подобных», при котором устраняются повторения среди строк. Эти матрицы представляют одну и ту же булеву функцию. Например, приведенная выше булева матрица оказывается эквивалентной троичной матрице:

$$\begin{array}{ccc} x_1 & x_2 & x_3 \\ - & - & 1 \\ 0 & 1 & - \end{array}$$

которая позволяет представить ту же булеву функцию $f(x_1, x_2, x_3)$, но более компактно.

Задание булевой функции троичной матрицей называется также *интервальным*.

Следует отметить неоднозначность представления булевой функции троичной матрицей – для одной и той же булевой матрицы существует в общем случае не одна эквивалентная ей троичная матрица. Две троичные матрицы считаются *эквивалентными*, если они эквивалентны одной и той же булевой матрице и соответственно представляют одну и ту же булеву функцию.

4. *Векторное задание*. При фиксированном упорядочении наборов значений аргументов (как правило, они упорядочиваются в соответствии с их позиционными двоичными кодами) булева функция полностью определяется второй частью ее табличного задания – вектор-столбцом ее значений. Булева функция n аргументов задается посредством 2^n-компонентного булева вектора.

Например, рассмотренная выше булева функция $f(x_1, x_2, x_3)$ (см. таблицу 6.1) представляется вектором 0 1 1 1 0 1 0 1, показывающим, что функция принимает значение 0 на наборах 0 0 0, 1 0 0, 1 1 0 и значение 1 на наборах 0 0 1, 0 1 0, 0 1 1, 1 0 1, 1 1 1. Очевидно, что вектор 0 1 1 1 0 1 0 1 совпадает с правым столбцом таблицы истинности функции (см. таблицу 6.1).

Векторная форма задания представляется достаточно компактной, когда число аргументов функции невелико.

5. *Алгебраическое задание*. Булева функция может быть представлена в виде алгебраического выражения суперпозиции элементарных логических операций, т. е. в виде формулы. Частными случаями таких представлений являются дизъюнктивные и конъюнктивные нормальные формы.

Например, приведенная выше булева функция $f(x_1, x_2, x_3)$ (см. таблицу 6.1) может быть представлена следующими совершенными дизъюнктивной и конъюнктивной нормальными формами:

$$f(\boldsymbol{x}) = \bar{x}_1\,\bar{x}_2\,x_3 \vee \bar{x}_1\,x_2\,\bar{x}_3 \vee \bar{x}_1\,x_2\,x_3 \vee x_1\,\bar{x}_2\,x_3 \vee x_1\,x_2\,x_3;$$

$$f(\boldsymbol{x}) = (x_1 \vee x_2 \vee x_3)(\bar{x}_1 \vee x_2 \vee x_3)(\bar{x}_1 \vee \bar{x}_2 \vee x_3).$$

6. *Графическое задание на кубе*. На гиперкубе булева функция $f(x_1, x_2, ..., x_n)$ может быть задана выделением вершин, представляющих элементы ее характеристического множества M_f^1. Так, на рисунке 6.10 затемнены вершины, соответствующие элементам множества M_f^1 рассматриваемой функции $f(\boldsymbol{x}) = \bar{x}_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 \bar{x}_3 \vee \bar{x}_1 x_2 x_3 \vee x_1 \bar{x}_2 x_3 \vee x_1 x_2 x_3$.

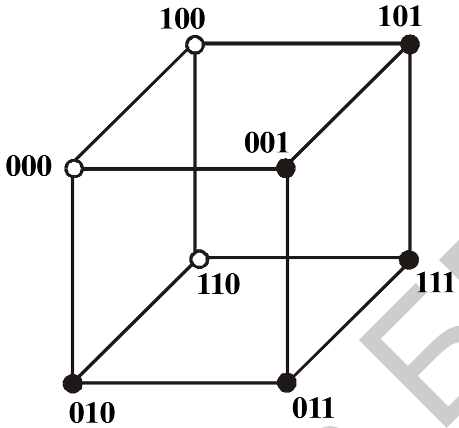


Рисунок 6.10 – Трехмерный куб с заданной на нем булевой функцией $f(\boldsymbol{x}) = \bar{x}_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 \bar{x}_3 \vee \bar{x}_1 x_2 x_3 \vee x_1 \bar{x}_2 x_3 \vee x_1 x_2 x_3$

При таком представлении интервалы характеристического множества M_f^1 получают некоторую наглядность, поскольку они соответствуют подкубам рассматриваемого гиперкуба и их можно распознавать визуально, глядя на графическое изображение булевой функции.

7. *Графическое задание на карте Карно*. На карте Карно булева функция $f(x_1, x_2, ..., x_n)$ задается указанием в каждой клетке значения, которое она принимает на соответствующем элементе булева пространства. Полностью определенная булева функция может быть задана также путем выделения тех клеток, которые соответствуют элементам булева пространства ее характеристического множества M_f^1. В этих клетках может проставляться, например, символ 1, жирная точка или они могут затемняться.

Например, на рисунке 6.11 таким образом задана рассмотренная выше функция $f(x_1, x_2, x_3) = \bar{x}_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 \bar{x}_3 \vee \bar{x}_1 x_2 x_3 \vee x_1 \bar{x}_2 x_3 \vee x_1 x_2 x_3$.

В общем случае, когда функция может быть и частичной, чтобы задать графически булеву функцию $f(x_1, x_2, ..., x_n)$, представленную в табличной форме (или в виде совершенной ДНФ), нужно построить n-мерную карту Карно и записать 1 в клетки, которые соответствуют элементам булева пространства E^n, представляющим наборы из M_f^1, и 0 – в клетки, соответствующие элементам булева пространства E^n, представляющим наборы из M_f^0.

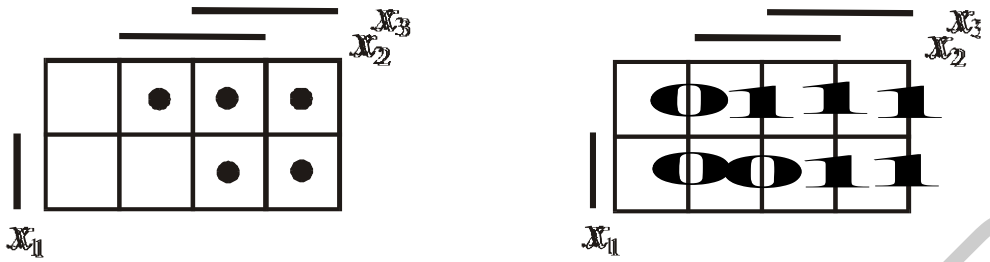


Рисунок 6.11 – Карты Карно булевой функции $f(x_1, x_2, x_3)$

6.2.3 Элементарные булевы функции

Поскольку любая булева функция от n аргументов определена на 2^n наборах значений ее аргументов, то число всевозможных булевых функций от n аргументов совпадает с числом всех 2^n-компонентных булевых векторов и равно 2^{2^n}. С ростом n эта величина быстро растет, принимая последовательно значения 4, 16, 256, 65 536, 4 294 967 296, … при n = 1, 2, 3, 4, 5, … .

Любую функцию от n аргументов можно считать функцией и от большего числа аргументов. Множество аргументов некоторой полностью определенной булевой функции $f(x_1, x_2, \ldots, x_n)$ можно однозначно разбить на два класса: *существенные аргументы,* от которых функция действительно зависит, и *несущественные* аргументы, изменение значений которых не приводит к изменению значения функции. Переменная x_i называется *существенным* аргументом функции $f(x_1, x_2, \ldots, x_n)$, если

$$f(x_1, \ldots, x_{i-1}, 1, x_i, \ldots, x_n) \neq f(x_1, \ldots, x_{i-1}, 0, x_i, \ldots, x_n),$$

в противном случае переменная x_i называется *несущественным* или фиктивным аргументом. В первом случае говорят также, что функция существенно зависит от аргумента x_i.

Несущественные аргументы можно исключить из представления функции, при этом получается функция, равносильная исходной. Например, рассмотрим функции

$$f(x, y, z) = (x\,\bar{y} \vee z)y, \;\; g(x, y, z) = x \vee y \vee \bar{x}\,\bar{y} \vee z.$$

Для функции $f(x, y, z)$ несущественным аргументом является x, для $g(x, y, z)$ – z. Исключение этих переменных из формул, представляющих функции, не изменит последние: $f(y, z) = (\bar{y} \vee z)y = yz$, $g(x, y) = x \vee y \vee \bar{x}\,\bar{y}$.

В число 2^{2^n} различных булевых функций от n аргументов входят также и функции, существенно зависящие от меньшего, чем n, числа аргументов.

Элементарными булевыми функциями являются функции от одного и двух аргументов. Всего имеется $2^2 = 4$ булевы функции одной переменной (таблица 6.2). Три из них тривиальны: функции f_0 и f_3 – функции-константы 0 и 1 (их значения не зависят от значения переменной x), функция f_1 повторяет значение переменной x: $f_1(x) = x$. Единственной нетривиальной функцией является $f_2(x)$ – одноместная булева функция, называемая *отрицанием* или *инверсией* аргумента x (ее значение всегда противоположно значению аргумента x).

Таблица 6.2 – Булевы функции от одного аргумента

x	0	1
$f_0 = 0$	0	0
$f_1 = x$	0	1
$f_2 = \bar{x}$	1	0
$f_3 = 1$	1	1

В таблице 6.3 приведены $2^{2^2} = 16$ булевых функций $f_i(x, y)$ от двух аргументов. В левом столбце показаны соответствующие им алгебраические представления в терминах нескольких функций, принятых за основные. В четырех правых столбцах приведены значения, принимаемые данными функциями на каждом из четырех наборов значений аргументов x и y.

Использование интервалов для представления области задания булевой функции приводит к рассмотрению троичных переменных. Введем серию операций над троичными переменными, определив их как такое обобщение соответствующих булевых операций, при котором значение «–» интерпретируется как символ неопределенности двоичного значения, т. е. считается, что рассматриваемая переменная принимает одно из двух значений: 0 или 1, но какое именно – неизвестно. Результат такого обобщения представлен в таблице 6.4, определяющей основные операции для разных наборов значений переменных (всего 3^2 различных набора значений из множества {0, 1, «–»}).

Таблица 6.3 – Булевы функции от двух переменных $f_i(x, y)$

Обозначение	Наименование	x: 0 y: 0	0 1	1 0	1 1
$f_0 = 0$	константа 0	0	0	0	0
$f_1 = x \wedge y$	конъюнкция	0	0	0	1
$f_2 = \overline{f}_{13}$	отрицание импликации $x \to y$	0	0	1	0
$f_3 = x$	значение переменной x	0	0	1	1
$f_4 = \overline{f}_{11}$	отрицание обратной импликации $y \to x$	0	1	0	0
$f_5 = y$	значение переменной y	0	1	0	1
$f_6 = x \oplus y$	дизъюнкция с исключением	0	1	1	0
$f_7 = x \vee y$	дизъюнкция	0	1	1	1
$f_8 = x \uparrow y = \overline{x \vee y}$	стрелка Пирса	1	0	0	0
$f_9 = x \sim y$	эквиваленция	1	0	0	1
$f_{10} = \overline{y}$	отрицание переменной y	1	0	1	0
$f_{11} = y \to x$	обратная импликация $y \to x$	1	0	1	1
$f_{12} = \overline{x}$	отрицание переменной x	1	1	0	0
$f_{13} = x \to y$	импликация $x \to y$	1	1	0	1
$f_{14} = x \mid y = \overline{x\,y}$	штрих Шеффера	1	1	1	0
$f_{15} = 1$	константа 1	1	1	1	1

Например, рассматривая дизъюнкцию $x \vee y$, полагаем, что эта функция принимает:

– значение 1, если известно, что по крайней мере одна из переменных x и y обладает значением 1;

– значение 0, если известно, что обе переменные имеют значение 0;

– значение «–» (т. е. остается неопределенной) во всех остальных случаях.

Нетрудно убедиться, что при таком обобщении на случай троичных операций имеет место прежняя интерпретация операций дизъюнкции и конъюнкции:

$$x \vee y = \max(x, z), \quad x \wedge y = \min(x, z),$$

считая, что $0 \le$ «–» и «–» ≤ 1. Сохраняются также остальные свойства всех операций и отношения между ними.

Таблица 6.4 – Определение основных операций
над троичными переменными

Аргумент $\boldsymbol{x}$ Аргумент $\boldsymbol{y}$	0 0 0 – – – 1 1 1 0 – 1 0 – 1 0 – 1
Отрицание $\bar{\boldsymbol{x}}$	1 1 1 – – – 0 0 0
Дизъюнкция $\boldsymbol{x} \vee \boldsymbol{y}$	0 – 1 – – 1 1 1 1
Конъюнкция $\boldsymbol{x} \wedge \boldsymbol{y}$ или $\boldsymbol{x}\,\boldsymbol{y}$	0 0 0 0 – – 0 – 1
Дизъюнкция с исключением $\boldsymbol{x} \oplus \boldsymbol{y}$	0 – 1 – – – 1 – 0
Эквиваленция $\boldsymbol{x} \sim \boldsymbol{y}$	1 – 0 – – – 0 – 1
Импликация $\boldsymbol{x} \to \boldsymbol{y}$	1 1 1 – – 1 0 – 1
Функция Пирса $\boldsymbol{x} \circ \boldsymbol{y} = \neg(\boldsymbol{x} \vee \boldsymbol{y})$	1 – 0 – – 0 0 0 0
Функция Шеффера $\boldsymbol{x} \mid \boldsymbol{y} = \neg(\boldsymbol{x} \wedge \boldsymbol{y})$	1 1 1 1 – – 1 – 0

Булевы операции отрицания, дизъюнкции и конъюнкции часто называются также операциями НЕ, ИЛИ и И соответственно (или NOT, OR и AND). Они образуют так называемый *булев базис*, алгебраические выражения с ними называют *булевыми формулами*, и этими формулами можно представить любую булеву функцию.

6.2.4 Теоретико-множественная интерпретация операций над булевыми функциями

Интерпретация операций над векторами. Поставим совокупность компонент булева вектора $\boldsymbol{x}$ во взаимно однозначное соответствие с некоторым конечным множеством A, число элементов которого равно длине вектора (как это делалось в пункте 1.1.2). Это означает, что каждой из компонент вектора $\boldsymbol{x}$ будет отвечать ровно один элемент множества A и, наоборот, каждому из элементов множества $a_i \in A$ будет отвечать только одна из компонент x_i вектора $\boldsymbol{x}$. При этом конкретным значениям булева вектора $\boldsymbol{x}$ соответствуют определенные подмножества $A_i \subseteq A$, образуемые теми элементами множества A, которым соответствуют компоненты вектора $\boldsymbol{x}$ со значением 1. Будем говорить, что именно эти подмножества представляются (задаются) значениями булева вектора $\boldsymbol{x}$.

Например, если вектор $\boldsymbol{x}_i$ имеет значение 1 0 0 1 0 1 0 1, то он представляет подмножество $A_i = \{a_1, a_4, a_6, a_8\}$ восьмиэлементного множества A. Пустому подмножеству $A_i = \varnothing$ соответствует вектор $\boldsymbol{x}_i$ = 0 0 0 0 0 0 0 0.

Операциям над множествами, в свою очередь, соответствуют логические операции над булевыми векторами, их представляющими. В частности, существует соответствие между операциями дизъюнкции, конъюнкции и инверсии над n-компонентными булевыми векторами, с одной стороны, и теоретико-множественными операциями объединения, пересечения и дополнения над n-элементными множествами, с другой стороны. Аналогично тому, как при выполнении теоретико-множественных операций над подмножествами $A_i \subseteq A$ вопрос о вхождении (или невхождении) в результат каждого из элементов множества A решается независимо, так и выполнение логических операций над булевыми векторами сводится к выполнению этих операций для каждой из n компонент векторов в отдельности. Логические операции над векторами являются, таким образом, покомпонентными (или поразрядными) операциями.

Пусть вектор $\boldsymbol{x} = (x_1, x_2, ..., x_n)$ представляет своими значениями $\boldsymbol{x}_i$ подмножество $A_i \subseteq A$. Тогда дополнение множества A_i представляется инверсией вектора $\boldsymbol{x}_i$ – вектором $\bar{\boldsymbol{x}}_i$, значение каждой компоненты которого есть инверсия значения соответствующей компоненты вектора $\boldsymbol{x}_i$. Операции инверсии булева вектора соответствует теоретико-множественная операция дополнения.

Действительно, если вектору $\boldsymbol{x}$ поставлено в соответствие множество $A = \{a_1, a_2, \dots, a_n\}$ и значением $\boldsymbol{x}_i = 0110000 1$ вектора $\boldsymbol{x}$ представляется подмножество $A_i = \{a_2, a_3, a_8\}$, то значением $1001111 0$ вектора $\bar{\boldsymbol{x}}_i$, получаемого в результате инверсии вектора $\boldsymbol{x}_i$, представляется подмножество $\bar{A}_i = \{a_1, a_4, a_5, a_6, a_7\}$, содержащее те и только те элементы множества A, которые отсутствуют в подмножестве A_i.

Операциям конъюнкция и дизъюнкция булевых векторов $\boldsymbol{x}_i$ и $\boldsymbol{x}_j$ соответствуют теоретико-множественные операции пересечения и объединения.

Например, если

$$\boldsymbol{x}_i = 11001100;$$
$$\boldsymbol{x}_j = 10101010,$$

то их конъюнкция

$$\boldsymbol{x}_i \wedge \boldsymbol{x}_j = 10001000$$

представляет множество $A_i \cap A_j = \{a_1, a_5\}$, являющееся пересечением множеств $A_i = \{a_1, a_2, a_5, a_6\}$ и $A_i = \{a_1, a_3, a_5, a_7\}$, а их дизъюнкция

$$\boldsymbol{x}_i \vee \boldsymbol{x}_j = 11101110$$

представляет множество $A_i \cup A_j = \{a_1, a_2, a_3, a_5, a_6, a_7\}$, являющееся объединением множеств A_i и A_i .

Аналогично определяются логические операции дизъюнкции с исключением, импликации и эквиваленции. При тех же значениях векторов $\boldsymbol{x}_i$ и $\boldsymbol{x}_j$

$$\boldsymbol{x}_i \oplus \boldsymbol{x}_j = 0\,1\,1\,0\,0\,1\,1\,0;$$
$$\boldsymbol{x}_i \rightarrow \boldsymbol{x}_j = 1\,0\,1\,1\,1\,0\,1\,1;$$
$$\boldsymbol{x}_i \sim \boldsymbol{x}_j = 1\,0\,0\,1\,1\,0\,0\,1.$$

Интерпретация операций над булевыми функциями. Операциям над булевыми функциями, в свою очередь, соответствуют операции над их характеристическими множествами (множествами единичных значений). В частности, существует определенное соответствие между булевыми функциями – дизъюнкция, конъюнкция и инверсия, с одной стороны, и теоретико-множественными операциями объединения, пересечения и дополнения их характеристических множеств, с другой стороны.

Пусть $f(\boldsymbol{x})$ и $g(\boldsymbol{x})$ – произвольные булевы функции от переменных $x_1, x_2, ..., x_n$, а M_f^1 и M_g^1 – их характеристические множества. Тогда дизъюнкция этих функций $f(\boldsymbol{x}) \vee g(\boldsymbol{x})$ есть функция, представляемая множеством $M_f^1 \cup M_g^1$. Конъюнкция $f(\boldsymbol{x}) \wedge g(\boldsymbol{x})$ есть функция, представляемая множеством $M_f^1 \cap M_g^1$, а инверсия любой из этих функций, т. е. $\bar{f}(\boldsymbol{x})$ (или $\bar{g}(\boldsymbol{x})$), представляется множеством $M \setminus M_f^1$ (или $M \setminus M_g^1$). Таким образом, имеем

$$M^1{}_{f\vee g} = M_f^1 \cup M_g^1;$$
$$M^1{}_{f\wedge g} = M_f^1 \cap M_g^1;$$
$$M^1{}_{\bar{f}} = M \setminus M_f^1;$$
$$M^1{}_{f\oplus g} = M_f^1 + M_g^1;$$
$$M^1{}_{f\rightarrow g} = M_g^1 \cup M_f^0;$$
$$M^1{}_{f\sim g} = M^0{}_{f\oplus g}.$$

В частности, можно таким образом непосредственно производить логические операции над функциями, заданными в матричном виде, используя теоретико-множественные операции над их характеристическими множествами. В то же время логические операции над функциями, заданными в векторном виде, сводятся к покомпонентным операциям над задающими их векторами.

Например, конъюнкция булевых функций $f(\boldsymbol{x})$ и $g(\boldsymbol{x})$, характеристические множества M_f^1 и M_g^1 которых заданы булевыми матрицами, может быть найдена как $M_f^1 \cap M_g^1$:

$$f(\boldsymbol{x}) = \begin{matrix} 0\,0\,1 \\ 0\,1\,0 \\ 1\,0\,1 \\ 1\,1\,0 \\ 1\,1\,1 \end{matrix} \qquad g(\boldsymbol{x}) = \begin{matrix} 0\,0\,0 \\ 0\,0\,1 \\ 0\,1\,1 \\ 1\,0\,1 \end{matrix} \qquad M^1{}_{f\wedge g} = \begin{matrix} 0\,0\,1 \\ 1\,0\,1 \end{matrix}$$

6.2.5 Векторные вычисления булевых функций

Если булевы функции заданы в векторном виде, то логические операции над ними можно выполнять аналогично операциям над задающими их булевыми векторами:

$$
\begin{array}{lcl}
f & = & 0\,1\,1\,0\,0\,1\,1\,0; \\
g & = & 1\,1\,0\,1\,0\,1\,0\,1; \\
f \wedge g & = & 0\,1\,0\,0\,0\,1\,0\,0; \\
f \vee g & = & 1\,1\,1\,1\,0\,1\,1\,1; \\
\bar{f} & = & 1\,0\,0\,1\,1\,0\,0\,1; \\
f \oplus g & = & 1\,0\,1\,1\,0\,0\,1\,1; \\
f \sim g & = & 0\,1\,0\,0\,1\,1\,0\,0; \\
f \to g & = & 1\,1\,0\,1\,1\,1\,0\,1.
\end{array}
$$

Вычисление произвольной композиции логических операций также можно свести к векторным вычислениям. Пусть булева функция задана в виде формулы (алгебраического выражения суперпозиции логических операций). Для установления порядка выполнения операций в формулах обычно используются скобки. Для выполнения вычислений удобным является бесскобочный вид формулы, позволяющий систематизировать процесс вычислений, – бесскобочная *форма Лукасевича*, или бесскобочная польская запись. Она была использована в пункте 5.1.3 при вычислении значений формул. Процедура перехода от формулы к бесскобочной форме Лукасевича поясняется на следующем примере. Например, структура логической формулы

$$f(x, y, z) = (x \vee \bar{y} z) \sim (\bar{x} \to z)$$

становится более явной при вынесении символов операций за скобки:

$$f(x, y, z) = \sim (\vee (x, \wedge (\neg(y), z)), \to (\neg(x), z)).$$

Удалив скобки и запятые, получим бесскобочную форму Лукасевича:

$$f(x, y, z) = \sim \vee x \wedge \neg y z \to \neg x z.$$

Бесскобочная форма интерпретируется далее как программа, которая читается справа налево и выполняется на одностороннем стеке (или магазине).

Если очередной символ представляет переменную, то представляющий ее булев вектор заносится в стек, в конец, называемый *вершиной стека*. При этом переменные *x, y, z* рассматриваются формально как простейшие функции от этих трех переменных и представляются соответствующими векторами ***x***, ***y*** и ***z*** длиной 2^3 (задающими своими одноименными компонентами все возможные комбинации значений трех переменных):

$$\begin{array}{ll} x & 0\,0\,0\,0\,1\,1\,1\,1 \\ y & 0\,0\,1\,1\,0\,0\,1\,1 \\ z & 0\,1\,0\,1\,0\,1\,0\,1\,. \end{array}$$

Если очередной символ оказывается оператором ¬, то выполняется соответствующая операция над последним элементом стека и этот элемент замещается результатом операции. В случае когда очередной символ оказывается двухместным оператором, соответствующая операция выполняется над двумя последними элементами стека, после чего они удаляются, а в стек заносится результат операции. Таким образом, число выполняемых операций равно длине бесскобочной формы.

Продемонстрируем этот процесс на данном примере. Содержимое стека в процессе вычислений будет меняться в следующей последовательности (для каждой операции показывается последний элемент стека):

$$\begin{array}{lll} z & 0\,1\,0\,1\,0\,1\,0\,1 & \\ x & 0\,0\,0\,0\,1\,1\,1\,1 & \\ \neg & 1\,1\,1\,1\,0\,0\,0\,0 & \bar{x} \\ \rightarrow & 0\,1\,0\,1\,1\,1\,1\,1 & \bar{x} \rightarrow z \\ z & 0\,1\,0\,1\,0\,1\,0\,1 & \\ y & 0\,0\,1\,1\,0\,0\,1\,1 & \\ \neg & 1\,1\,0\,0\,1\,1\,0\,0 & \bar{y} \\ \wedge & 0\,1\,0\,0\,0\,1\,0\,0 & \bar{y}z \\ x & 0\,0\,0\,0\,1\,1\,1\,1 & \\ \vee & 0\,1\,0\,0\,1\,1\,1\,1 & x \vee \bar{y}z \\ \sim & 1\,1\,1\,0\,1\,1\,1\,1 & (x \vee \bar{y}z) \sim (\bar{x} \rightarrow z)\,. \end{array}$$

Полученный вектор 1 1 1 0 1 1 1 1 представляет результат вычисления функции $f(x,y,z)$:

$$M_f^1 = \{000, 001, 010, 100, 101, 110, 111\},\ \ M_f^0 = \{011\}.$$

Так же как и в случае применения польской записи для представления формул алгебры логики (пункт 5.1.3), приведенные выше вычисления можно описать в виде обхода соответствующего дерева.

6.3 Некоторые важные классы булевых функций

6.3.1 Двойственные функции

Булева функция $f^*(x_1,x_2,\ldots,x_n)$, равная $\bar{f}(\bar{x}_1, \bar{x}_2,\ldots, \bar{x}_n)$, называется *двойственной* функцией по отношению к функции $f(x_1,x_2,\ldots,x_n)$. Для получения таблицы истинности функции $f^*(x_1, x_2, \ldots, x_n)$ достаточно в таблице истинности функции $f(x_1, x_2, \ldots, x_n)$ заменить значения всех переменных на противоположные, т. е. все единицы (во всех столбцах и строках) заменить на нули, а нули – на единицы. Но так как в таблице истинности наборы значений аргументов лексико-графически упорядочены (по возрастанию представляемых ими двоичных чисел), то столбец значений $f^*(x_1, x_2, \ldots, x_n)$ может быть получен простым инвертированием столбца значений функции $f(x_1, x_2, \ldots, x_n)$ и его переворачиванием, как это показано в таблице 6.4.

Легко видеть, что:

– константа 0 двойственна 1;

– константа 1 двойственна 0;

– функция x двойственна x (так как $\neg(\bar{x}) = x$);

– функция $\bar{x}$ двойственна $\bar{x}$ (так как $\bar{x} = \bar{x}$);

– функция $x_1 x_2$ двойственна $x_1 \vee x_2$ (так как $\overline{\bar{x}\,\bar{y}} = x_1 \vee x_2$);

– функция $x_1 \vee x_2$ двойственна $x_1 x_2$ (так как $\overline{\bar{x} \vee \bar{y}} = x_1 x_2$).

Таблица 6.4 – Задание функций f и f^*

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

Из определения двойственности вытекает, что функция f является двойственной по отношению к функции f^*:

$$(f^*)^* = f^{**} = f.$$

Функция, двойственная суперпозиции некоторых функций, равносильна соответствующей суперпозиции двойственных функций.

Действительно, если функция $f(x_1, x_2, \ldots, x_n)$ выражена формулой

$$\Phi = g\,[g_1(x_{11}, x_{12}, \ldots, x_{1n1}), \ldots, g_m(x_{m1}, x_{m2}, \ldots, x_{mkm})],$$

то формула для двойственной ей функции получается следующим образом:

$$
\begin{aligned}
& f^*(x_1, x_2, \ldots, x_n) = \overline{f}(\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n) = \\
& = \overline{g}[g_1(\overline{x}_{11}, \overline{x}_{12}, \ldots, \overline{x}_{1n1}), \ldots, g_m(\overline{x}_{m1}, \overline{x}_{m2}, \ldots, \overline{x}_{mkm})] = \\
& = \overline{g}[\overline{\overline{g}}_1(\overline{x}_{11}, \overline{x}_{12}, \ldots, \overline{x}_{1n1}), \ldots, \overline{\overline{g}}_m(\overline{x}_{m1}, \overline{x}_{m2}, \ldots, \overline{x}_{mkm})] = \\
& = \overline{g}[\overline{g}_1^*(x_{11}, x_{12}, \ldots, x_{1n1}), \ldots, \overline{g}_m^*(x_{m1}, x_{m2}, \ldots, x_{mkm})] = \\
& = g^*[g_1^*(x_{11}, x_{12}, \ldots, x_{1n1}), \ldots, g_m^*(x_{m1}, x_{m2}, \ldots, x_{mkm})].
\end{aligned} \tag{6.1}
$$

Функция $f(x_1, x_2, \ldots, x_n)$ называется *самодвойственной*, если она равна двойственной себе: $f = f^*$:

$$f(x_1, x_2, \ldots, x_n) = \overline{f}(\overline{x}_1, x_1, \overline{x}_2, \ldots, \overline{x}_n).$$

Например, функция, заданная таблицей 6.4, не является самодвойственной, так как $f(x_1, x_2, x_3) \neq f^*(x_1, x_2, x_3)$.

Из определения самодвойственности следует, что такая функция на каждой паре противоположных наборов принимает противоположные значения. Например, если $f(0, 1, 1) = 0$, то $f(1, 0, 0) = 1$. Это условие не выполняется для функции $f(x_1, x_2, x_3)$ (таблица 6.4), что означает, что она не является самодвойственной.

Из определения самодвойственности следует, что такая функция полностью определяется своими значениями на первой половине строк таблицы истинности, если наборы значений аргументов упорядочены по возрастанию представляемых ими двоичных чисел. Для проверки функции на самодвойственность по таблице истинности достаточно инвертировать столбец значений этой функции и сравнить его с исходным. Равенство столбцов говорит о том, что анализируемая функция является самодвойственной.

Самодвойственными функциями являются, например, функции x и $\overline{x}$, не самодвойственны константы 0 и 1, $x \wedge y$, $x \vee y$, $x \oplus y$, $x \mid y$.

6.3.2 Принцип двойственности

Из (6.1) вытекает *принцип двойственности*, который формулируется следующим образом.

Если формула $\Phi = g[g_1, g_2, \ldots, g_m]$ задает функцию $f(x_1, x_2, \ldots, x_n)$, то формула $g[g_1^*, g_2^*, \ldots, g_m^*]$, полученная из Φ заменой функций g_i на двойственные g_i^*, задает функцию $f^*(x_1, x_2, \ldots, x_n)$, двойственную $f(x_1, x_2, \ldots, x_n)$.

Формула $g[g_1^*, g_2^*, \ldots, g_m^*]$ называется *двойственной* формуле $\Phi = g[g_1, g_2, \ldots, g_m]$. Таким образом, $\Phi^* = g[g_1^*, g_2^*, \ldots, g_m^*]$.

Например, если $\Phi = \overline{x} \oplus yz \vee \overline{y}$, то

$$\Phi^* = \overline{(x \oplus \overline{y}\overline{z}) \vee y)} = (x \sim \overline{y}\ \overline{z})\ \overline{y} = (x\ \overline{y}\ \overline{z} \vee \overline{x} y \vee \overline{x} z)\ \overline{y} = x\ \overline{y}\ \overline{z} \vee \overline{x}\ \overline{y} z.$$

Принцип двойственности для булевых формул над множеством $F = \{0, 1, x_i, \bar{x}_i, \wedge, \vee\}$ конкретизируется следующим образом: если в булевой формуле Φ над F, представляющей функцию f, заменить 0 на 1, 1 на 0, дизъюнкцию на конъюнкцию, конъюнкцию на дизъюнкцию, то получим формулу Φ*, представляющую функцию, двойственную f.

Например, если $\Phi = (\bar{y} \vee \bar{x}z)(y \vee \bar{w})$, то $\Phi^* = \bar{y}(\bar{x} \vee z) \vee y\bar{w}$.

Из вышесказанного следует также, что функции, двойственные равносильным функциям, также равносильны. Таким образом, производя замену вхождений элементов $\{0, 1, \wedge, \vee\}$ на $\{1, 0, \vee, \wedge\}$ в равносильных формулах, получаем равносильные же формулы. Это проявляется в основных законах булевой алгебры, рассмотренных в пункте 5.3.1.

Кроме того, принцип двойственности может использоваться для равносильных преобразований формул.

6.3.3 Монотонные функции

Введем отношение *предшествования* «≤» (отношение частичного порядка) на множестве булевых векторов одинаковой длины ($\sigma_i, \tau_i \in \{0, 1\}$):

$$(\sigma_1, \sigma_2, \ldots, \sigma_n) \leq (\tau_1, \tau_2, \ldots, \tau_n),$$

если и только если $\sigma_i \leq \tau_i$ для всех $i = 1, 2, \ldots, n$, считая, что $0 < 1$. Следует отметить, что не любые пары наборов находятся в отношении предшествования.

Например, $(0, 1, 0, 1) \leq (1, 1, 0, 1)$, но $(0, 1, 0, 1)$ и $(1, 0, 0, 1)$ в отношении предшествования не находятся.

Функция $f(x_1, x_2, \ldots, x_n)$ называется *монотонной*, если для любых двух наборов $\boldsymbol{\sigma}$ и $\boldsymbol{\tau}$ значений аргументов длиной n, таких, что $\boldsymbol{\sigma} \leq \boldsymbol{\tau}$, имеет место неравенство $f(\boldsymbol{\sigma}) \leq f(\boldsymbol{\tau})$.

Монотонными являются, например, функции x, $x \wedge y$, $x \vee y$, константы 0 и 1, немонотонными – $\bar{x}$, $x \wedge \bar{y}$, $x \oplus y$. Можно заметить, что всякая немонотонная функция содержит в себе в каком-то виде функцию отрицания.

Для проверки функции на монотонность необходимо проверить выполнение неравенства $f(\boldsymbol{\sigma}) \leq f(\boldsymbol{\tau})$ для всех пар наборов, находящихся в отношении предшествования $\boldsymbol{\sigma} \leq \boldsymbol{\tau}$ (кроме $\boldsymbol{\sigma} = \boldsymbol{\tau}$).

Функция, заданная таблицей 6.4, не является монотонной, так как для пары наборов значений аргументов $(0\,0\,0) \leq (0\,1\,0)$ имеем $f(0, 0, 0) > f(0, 0, 0)$.

6.3.4 Линейные функции

Алгебра логики на основе двух операций – конъюнкции и дизъюнкции с исключением (суммы по модулю два) при наличии константы 1 – называется *алгеброй Жегалкина*. В этой алгебре любая булева функция представима полиномом Жегалкина. *Полиномом Жегалкина* называется дизъюнкция с исключением конечного числа различных элементарных конъюнкций переменных без знаков инверсий. Например, полиномами Жегалкина являются следующие выражения:

$$xyz \oplus xy \oplus x \oplus z \oplus 1; \quad x \oplus y \oplus z \oplus 1; \quad x \oplus 1; \quad xy; \quad z; \quad 1.$$

Подобно операциям дизъюнкции и конъюнкции, дизъюнкция с исключением обладает свойствами коммутативности и ассоциативности, а также обобщается на случай многих переменных.

Жегалкиным И. было доказано, что предложенный им полином является каноническим представлением булевой функции, т. е. любая булева функция (кроме константы 0) может быть представлена в виде полинома Жегалкина, притом единственным образом, и различные полиномы реализуют разные булевы функции.

Функция $f(x_1, x_2, \ldots, x_n)$ называется *линейной,* если ее можно представить линейным полиномом в алгебре Жегалкина:

$$f(x_1, x_2, \ldots, x_n) = a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus \ldots \oplus a_nx_n,$$

где все коэффициенты a_i равны 0 или 1.

Полином Жегалкина можно получить из СДНФ (а значит, и по таблице истинности) булевой функции. СДНФ состоит из взаимно ортогональных полных конъюнкций, т. е. на любом наборе значений переменных значение 1 может принять только одна из них. Поэтому многоместную дизъюнкцию, связывающую члены совершенной ДНФ, можно заменить многоместной суммой по модулю два. Но при этом необходимо избавиться от операций отрицания переменных, что возможно путем замены $\bar{x}$ на $x \oplus 1$ согласно равносильности $\bar{a} = a \oplus 1$.

Например, полином Жегалкина для СДНФ

$$f(a,b,c) = \bar{a}\,\bar{b}\,c \vee \bar{a}\,b\,c \vee a\,\bar{b}\,\bar{c}$$

получается следующим образом

$$f(a,b,c) = (a \oplus 1)(b \oplus 1)\,c \oplus (a \oplus 1)\,b\,c \oplus a\,(b \oplus 1)(c \oplus 1) =$$
$$= a\,b\,c \oplus a\,c \oplus b\,c \oplus c \oplus a\,b\,c \oplus b\,c \oplus a\,b\,c \oplus a\,b\,c \oplus a\,b \oplus a\,c \oplus a =$$
$$= (a\,b\,c \oplus a\,b\,c) \oplus (a\,b\,c \oplus a\,b\,c) \oplus (a\,c \oplus a\,c) \oplus (b\,c \oplus b\,c) \oplus c \oplus a\,b \oplus a =$$

$$= a\,b \oplus a \oplus c,$$

воспользовавшись следующими равносильностями:

$$a \oplus b = b \oplus a;\;\; a \oplus a = 0;\;\; a\,a = a;\;\; (a \oplus b)c = ac \oplus bc.$$

Полином Жегалкина можно построить и исходя из произвольной ДНФ, но для этого придется предварительно ортогонализовать ее элементарные конъюнкции, чтобы на любом наборе значений переменных значение 1 принимала только одна конъюнкция. Для ДНФ с небольшим числом элементарных конъюнкций при переходе к полиному Жегалкина можно воспользоваться также следующим соотношением:

$$A \vee B = A \oplus B \oplus AB,$$

где A и B – некоторые конънкции (или даже ДНФ). Например:

$$a\,b \vee \bar{a}\,\bar{b} = a\,b \oplus \bar{a}\,\bar{b} \oplus a\,b\,\bar{a}\,\bar{b} = a\,b \oplus \bar{a}\,\bar{b} = a\,b \oplus (a \oplus 1)(b \oplus 1) =$$

$$= a\,b \oplus a\,b \oplus a \oplus b \oplus 1 = a \oplus b \oplus 1.$$

Линейными функциями являются, например, x, $\bar{x}$, $x \oplus y$, константы 0 и 1. Нелинейными являются $x \wedge y$, $x \vee y$ (так как $x \vee y = x \oplus y \oplus xy$).

Функция, заданная таблицей 6.4, не является линейной, так как она представляется следующим полиномом Жегалкина, который является нелинейным:

$$f(x_1, x_2, x_3) = (x_1 \oplus 1)\,(x_2 \oplus 1)\,(x_3 \oplus 1) \oplus (x_1 \oplus 1)\,(x_2 \oplus 1)\,x_3 \oplus x_1\,(x_2 \oplus 1)\,x_3 =$$

$$= x_1x_2x_3 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1 \oplus x_2 \oplus 1,$$

который легко получается из совершенной дизъюнктивной нормальной формы этой функции $f(x_1,x_2,x_3) = \bar{x}_1\,\bar{x}_2\,\bar{x}_3 \vee \bar{x}_1\,\bar{x}_2\,x_3 \vee x_1\,\bar{x}_2\,x_3$ путем замены символа операции «$\vee$» на «$\oplus$», инверсии $\bar{x}$ переменной на $x \oplus 1$.

6.4 Разложение булевых функций по переменным

6.4.1 Дизъюнктивное разложение Шеннона

Теорема Шеннона. Любая булева функция $f(x_1, x_2, \ldots, x_n)$ при любом m $(1 \le m \le n)$ может быть представлена в следующем виде:

$$f(x_1, x_2, \ldots, x_n) = \bigvee_{\sigma_1,\sigma_2,\ldots,\sigma_m} x_1^{\sigma_1} x_2^{\sigma_2} \ldots x_m^{\sigma_m} f(\sigma_1, \sigma_2, \ldots, \sigma_m, x_{m+1}, \ldots, x_n), \qquad (6.2)$$

где дизъюнкция берется по всевозможным 2^m наборам значений некоторых (не обязательно взятых подряд) переменных x_1, x_2, …, x_m, где $\sigma_n \in \{0,1\}$, $x^\sigma = x$ при $\sigma = 1$ и $x^\sigma = \bar{x}$ при $\sigma = 0$.

Представление (6.2) называется *дизъюнктивным разложением функции* по переменным $x_1, x_2, \ldots, x_m$. Функции $f(\sigma_1, \sigma_2, \ldots, \sigma_m, x_{m+1}, \ldots, x_n)$ называются коэффициентами разложения функции $f(x_1, x_2, \ldots, x_n)$ по m переменным. Они получаются в результате подстановки констант $\sigma_1, \sigma_2, \ldots, \sigma_n$ вместо вхождений переменных $x_1, x_2, \ldots, x_m$ и не зависят от переменных $x_1, x_2, \ldots, x_m$.

Для доказательства теоремы подставим в обе части равенства (6.2) произвольный набор $(\alpha_1, \alpha_2, \ldots, \alpha_n)$ значений всех n переменных. Так как $x^\sigma = 1$ только при $x = \sigma$, то из 2^m конъюнкций $x_1^{\sigma_1} x_2^{\sigma_2} \ldots x_m^{\sigma_m}$ правой части (6.2) значение 1 примет единственная конъюнкция – та, для которой $\sigma_1 = \alpha_1$, $\sigma_2 = \alpha_2$, …, $\sigma_m = \alpha_m$. Остальные конъюнкции будут равны 0. Отсюда получим тождество

$$f(\alpha_1, \alpha_2, \ldots, \alpha_n) = \alpha_1^{\alpha_1} \alpha_2^{\alpha_2} \ldots \alpha_m^{\alpha_m} f(\alpha_1, \alpha_2, \ldots, \alpha_m, \alpha_{m+1}, \ldots, \alpha_n).$$

В качестве примера разложим следующую функцию:

$$f(v,w,x,y,z) = \bar{v}\,\bar{w}\,\bar{x}\,\bar{y}\,\bar{z} \vee v\,\bar{x}\,\bar{y}z \vee v\,\bar{x}y\,\bar{z} \vee vx\,\bar{y}\,\bar{z} \vee vwz \vee \bar{v}\,\bar{w}$$

по переменным x и y:

$$f(v,w,x,y,z) = \bar{x}\,\bar{y}(\bar{v}\,\bar{w}\,\bar{z} \vee vz \vee vwz \vee \bar{v}\,\bar{w}) \vee \bar{x}y(v\,\bar{z} \vee vwz \vee \bar{v}\,\bar{w}) \vee$$

$$\vee x\,\bar{y}(v\,\bar{z} \vee vwz \vee \bar{v}\,\bar{w}) \vee xy(vwz \vee \bar{v}\,\bar{w}) =$$

$$= \bar{x}\,\bar{y}(vz \vee \bar{v}\,\bar{w}) \vee \bar{x}y(v\,\bar{z} \vee vw \vee \bar{v}\,\bar{w}) \vee x\,\bar{y}(v\,\bar{z} \vee vw \vee \bar{v}\,\bar{w}) \vee xy(vwz \vee \bar{v}\,\bar{w}).$$

Из теоремы Шеннона вытекают два частных случая разложения (6.2), справедливых для двух крайних значений числа m: $m = 1$ и $m = n$.

1. Любая булева функция $f(x_1, x_2, \ldots, x_n)$ может быть представлена в следующем виде (разложена по одной из своих переменных):

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

2. Любая булева функция $f(x_1, x_2, \ldots, x_n)$ может быть представлена в следующем виде (разложена по всем своим переменным):

$$f(x_1, x_2, \ldots, x_n) = \bigvee_{\sigma_1, \sigma_2, \ldots, \sigma_n} x_1^{\sigma_1} x_2^{\sigma_2} \ldots x_n^{\sigma_n} f(\sigma_1, \sigma_2, \ldots, \sigma_n). \qquad (6.3)$$

Разложение (6.3) является представлением булевой функции $f(x_1, x_2, \ldots, x_n)$ в СДНФ. Коэффициентами разложения (6.3) $f(\sigma_1, \sigma_2, \ldots, \sigma_n)$ являются константы 0 или 1 в зависимости от значения функции $f(x_1, x_2, \ldots, x_n)$ на наборе $(\sigma_1, \sigma_2, \ldots, \sigma_n)$.

Для примера найдем коэффициенты разложения следующей функции:

$$f(x, y, z) = y \oplus x\,\bar{z} \to \bar{y}z \tag{6.4}$$

по переменной x:

$$f(0, y, z) = y \oplus 0 \to \bar{y}z = y \to \bar{y}z;$$

$$f(1, y, z) = y \oplus \bar{z} \to \bar{y}z.$$

Далее найдем коэффициенты разложения этой же функции по двум переменным x и y (это может быть сделано и разложением коэффициентов $f(0, y, z)$ и $f(1, y, z)$ по переменной y):

$$f(0, 0, z) = 0 \to z = 1;$$

$$f(0, 1, z) = 1 \to 0 = 0;$$

$$f(1, 0, z) = 0 \oplus \bar{z} \to z = \bar{z} \to z = z;$$

$$f(1, 1, z) = 1 \oplus \bar{z} \to 0 = z \to 0 = \bar{z}.$$

С учетом полученных коэффициентов получается следующее разложение функции по переменным x и y:

$$f(x, y, z) = \bar{x}\,\bar{y} \vee x\,\bar{y}z \vee xy\,\bar{z}.$$

Разложение функции $f(x, y, z)$ по всем переменным приводит ее к виду совершенной ДНФ:

$$f(x, y, z) = \bar{x}\,\bar{y}\,\bar{z} \vee \bar{x}\,\bar{y}z \vee x\,\bar{y}z \vee xy\,\bar{z}.$$

В таблицах 6.5 и 6.6 приводится пример разложения функции $f(x_1,x_2,x_3,x_4)$ по переменным x_2 и x_3. Коэффициенты разложения $f(x_1,\sigma_2,\sigma_3,x_4)$ находятся путем выделения тех строк таблицы истинности, в которых $x_2 = \sigma_2$, $x_3 = \sigma_3$. В алгебраическом виде полученные коэффициенты разложения имеют вид

$$f(x_1, 0, 0, x_4) = \bar{x}_1\,\bar{x}_4 \vee \bar{x}_1 x_4; \quad f(x_1, 0, 1, x_4) = \bar{x}_1 x_4 \vee x_1\,\bar{x}_4 \vee x_1 x_4;$$

$$f(x_1, 1, 0, x_4) = \bar{x}_1\,\bar{x}_4; \quad f(x_1, 1, 1, x_4) = \bar{x}_1\,\bar{x}_4 \vee \bar{x}_1 x_4 \vee x_1 x_4.$$

Таблица 6.5 – Пример функции f

x_1	x_2	x_3	x_4	$f(x_1,x_2,x_3,x_4)$
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Таблица 6.6 – Коэффициенты разложения функции f

x_1	x_4	$f(x_1,0,0,x_4)$	$f(x_1,0,1,x_4)$	$f(x_1,1,0,x_4)$	$f(x_1,1,1,x_4)$
0	0	1	0	1	1
0	1	1	1	0	1
1	0	0	1	0	0
1	1	0	1	0	1

6.4.2 Конъюнктивное разложение Шеннона

В силу двойственности ДНФ и КНФ из выражения (6.2) получается следующее представление булевой функции $f(x_1, x_2, \ldots, x_n)$:

$$f(x_1, x_2,\ldots,x_n) = \bigwedge_{\sigma_1,\sigma_2,\ldots,\sigma_m} (x_1^{\bar{\sigma}_1} \vee x_2^{\bar{\sigma}_2} \vee \ldots \vee x_m^{\bar{\sigma}_m} \vee f(\sigma_1,\sigma_2,\ldots,\sigma_m, x_{m+1},\ldots,x_n)). \quad (6.5)$$

Это представление называется *конъюнктивным разложением функции* $f(x_1, x_2, \ldots, x_n)$ по переменным $x_1, x_2, \ldots, x_m$. Справедливость формулы (6.5) может быть формально доказана аналогично доказательству (6.2).

Из конъюнктивного разложения (6.5) функции также вытекают два частных случая: при $m = 1$ и $m = n$.

1. Любая булева функция $f(x_1, x_2, \ldots, x_n)$ может быть представлена в следующем виде (разложена по одной из своих переменных):

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

2. Любая булева функция $f(x_1, x_2, \ldots, x_n)$ может быть представлена в следующем виде (разложена по всем своим переменным):

$$f(x_1, x_2, \ldots, x_n) = \bigwedge_{\sigma_1, \sigma_2, \ldots, \sigma_n} (x_1^{\bar{\sigma}_1} \vee x_2^{\bar{\sigma}_2} \vee \ldots \vee x_n^{\bar{\sigma}_n} \vee f(\sigma_1, \sigma_2, \ldots, \sigma_n)). \qquad (6.6)$$

Разложение (6.6) является представлением булевой функции $f(x_1, x_2, \ldots, x_n)$ в СКНФ. Коэффициентами разложения (6.6) $f(\sigma_1, \sigma_2, \ldots, \sigma)$ являются константы 0 или 1 в зависимости от значения функции $f(x_1, x_2, \ldots, x_n)$ на наборе $(\sigma_1, \sigma_2, \ldots, \sigma)$.

Для примера найдем коэффициенты разложения функции $f(x, y, z) = y \oplus x\,\bar{z} \to \bar{y}z$ по переменной x:

$$f(1, y, z) = y \oplus \bar{z} \to \bar{y}z;$$

$$f(0, y, z) = y \to \bar{y}z,$$

что порождает следующее разложение:

$$f(x, y, z) = (\bar{x} \vee f(1, y, z))\,(x \vee f(0, y, z)).$$

Далее найдем коэффициенты разложения этой же функции по переменным x и y (это может быть сделано и разложением коэффициентов $f(1, y, z)$ и $f(0, y, z)$ по переменной y):

$$f(1, 1, z) = 1 \oplus \bar{z} \to 0 = z \to 0 = \bar{z};$$

$$f(1, 0, z) = 0 \oplus \bar{z} \to z = \bar{z} \to z = = z;$$

$$f(0, 1, z) = 1 \to = 0;$$

$$f(0, 0, z) = 0 \to z = 1,$$

что порождает следующее разложение:

$$f(x, y, z) = (\bar{x} \vee \bar{y} \vee f(1, 1, z))\,(\bar{x} \vee y \vee f(1, 0, z))\,(x \vee \bar{y} \vee f(0, 1, z))\,(x \vee y \vee f(0, 0, z)) =$$

$$= (\bar{x} \vee \bar{y} \vee \bar{z})\,(\bar{x} \vee y \vee z)\,(x \vee \bar{y}).$$

И наконец, разложение функции $f(x, y, z)$ по всем переменным приводит ее к виду совершенной КНФ:

$$f(x, y, z) = (\bar{x} \vee \bar{y} \vee \bar{z})\,(\bar{x} \vee y \vee z)\,(x \vee \bar{y} \vee \bar{z})\,(x \vee \bar{y} \vee z).$$

6.5 Системы булевых функций

На практике часто приходится иметь дело не с одной булевой функцией, а с их системой. Как правило, в систему объединяются функции, определенные на одном и том же булевом пространстве. Например, система $F = (f_1, f_2, ..., f_m)$ из m булевых функций, зависящих от n аргументов, имеет вид

$$y_1 = f_1(x_1, x_2, ..., x_n);$$

$$y_2 = f_2(x_1, x_2, ..., x_n);$$

$$\ldots \qquad \ldots$$

$$y_m = f_m(x_1, x_2, ..., x_n).$$

Ее удобно представлять как *векторную булеву функцию* $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, не забывая, что

$$\boldsymbol{x} = (x_1, x_2, ..., x_n),\ \ \boldsymbol{y} = (y_1, y_2, ..., y_m),\ \text{ а }\ \boldsymbol{f} = (f_1, f_2, ..., f_m).$$

Пусть система $F = (f_1, f_2, ..., f_m)$ состоит из частичных булевых функций $y_i = f_i(\boldsymbol{x})$, заданных соответственно характеристическими множествами $M_{fi}{}^1$ и $M_{fi}{}^0$, образующими области $M_{fi} = M_{fi}{}^1 \cup M_{fi}{}^0$ определения функций системы и представляющими собой некоторые подмножества общего булева пространства M. Тогда область определения системы в целом есть объединение областей определения ее функций:

$$M_F = M_{f1} \cup M_{f2} \cup \ldots \cup M_{fm}.$$

Очевидно, что проще всего задать эту систему, перечислив все наборы значений переменных $x_1, x_2, ..., x_n$, составляющие множество M_F, и соответствующие им значения функций $f_1, f_2, ... , f_m$. Совокупности соответствующих им значений векторов $\boldsymbol{x}$ и $\boldsymbol{y}$ можно собрать в булевы матрицы $\boldsymbol{X}$ и $\boldsymbol{Y}$. Столбцам этих матриц сопоставляются переменные $x_1, x_2, ..., x_n$ и $y_1, y_2, ..., y_m$ соответственно. Строками матрицы $\boldsymbol{X}$ служат векторы, представляющие наборы значений переменных $x_1, x_2, ..., x_n$, а соответствующими строками матрицы $\boldsymbol{Y}$ – векторы, представляющие наборы значений функций. Число строк матриц $\boldsymbol{X}$ и $\boldsymbol{Y}$ равно мощности области определения системы функций ($|M_F|$), а число их столбцов равно числу аргументов и числу функций соответственно.

Эта форма задания применима также и для представления системы полностью определенных булевых функций, а также таких неполностью определенных булевых функций, области определения которых совпадают. Например, пара булевых матриц

$$
\begin{array}{cccccccc}
 & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \\
 & 1 & 1 & 1 & 0 & 0 & 0 \\
 & 1 & 0 & 0 & 1 & 0 & 1 \\
\boldsymbol{X} = & 1 & 0 & 1 & 0 & 0 & 0 \\
 & 1 & 0 & 1 & 1 & 0 & 1 \\
 & 1 & 1 & 0 & 1 & 0 & 1 \\
 & 1 & 1 & 1 & 0 & 1 & 0
\end{array}
\qquad
\begin{array}{ccccc}
 & y_1 & y_2 & y_3 & \\
 & 1 & 1 & 1 & 1 \\
 & 1 & 0 & 0 & 2 \\
\boldsymbol{Y} = & 0 & 1 & 1 & 3 \\
 & 1 & 0 & 0 & 4 \\
 & 0 & 1 & 0 & 5 \\
 & 0 & 1 & 1 & 6
\end{array}
$$

может рассматриваться двояко:

– как задание системы из трех полностью определенных булевых функций от шести переменных:

$$y_1 = x_1 x_2 x_3 \bar{x}_4 \bar{x}_5 \bar{x}_6 \vee x_1 \bar{x}_2 \bar{x}_3 x_4 \bar{x}_5 x_6 \vee x_1 \bar{x}_2 x_3 x_4 \bar{x}_5 x_6;$$

$$y_2 = x_1 x_2 x_3 \bar{x}_4 \bar{x}_5 \bar{x}_6 \vee x_1 \bar{x}_2 x_3 \bar{x}_4 \bar{x}_5 \bar{x}_6 \vee x_1 x_2 \bar{x}_3 x_4 \bar{x}_5 x_6 \vee x_1 x_2 x_3 \bar{x}_4 x_5 \bar{x}_6;$$

$$y_3 = x_1 x_2 x_3 \bar{x}_4 \bar{x}_5 \bar{x}_6 \vee x_1 \bar{x}_2 x_3 \bar{x}_4 \bar{x}_5 \bar{x}_6 \vee \vee x_1 x_2 x_3 \bar{x}_4 x_5 \bar{x}_6,$$

при этом множество M_{fi}^0 каждой из трех функций f_i определяется как $M_{fi}^0 = M \setminus M_{fi}^1$;

– как задание системы из трех не полностью определенных булевых функций f_i, каждая из которых определена только на шести наборах значений переменных:

$$M_{f1}^1 = \{1\,1\,1\,0\,0\,0,\ 1\,0\,0\,1\,0\,1,\ 1\,0\,1\,1\,0\,1\},\ M_{f1}^0 = \{1\,0\,1\,0\,0\,0,\ 1\,1\,0\,1\,0\,1,\ 1\,1\,1\,0\,1\,0\};$$

$$M_{f2}^1 = \{1\,1\,1\,0\,0\,0,\ 1\,0\,1\,0\,0\,0,\ 1\,1\,0\,1\,0\,1,\ 1\,1\,1\,0\,1\,0\},\ M_{f2}^0 = \{1\,0\,0\,1\,0\,1,\ 1\,0\,1\,1\,0\,1\};$$

$$M_{f3}^1 = \{1\,1\,1\,0\,0\,0,\ 1\,0\,1\,0\,0\,0,\ 1\,1\,1\,0\,1\,0\},\ M_{f3}^0 = \{1\,0\,0\,1\,0\,1,\ 1\,0\,1\,1\,0\,1,\ 1\,1\,0\,1\,0\,1\},$$

на остальных наборах их значения не определены: $M_{fi}^- = M \setminus (M_{fi}^1 \cup M_{fi}^0)$.

Следует заметить, что такой парой булевых матриц можно задать лишь те системы частичных функций, у которых области определения функций совпадают, как для рассматриваемой системы функций:

$$M_{f1} = M_{f2} = M_{f3} = M_F, \quad M_{f1}^- = M_{f2}^- = M_{f3}^-.$$

Если же области определения функций системы не совпадают, матрицей $\boldsymbol{X}$ представляется объединение этих областей, а матрица $\boldsymbol{Y}$ становится троичной: наряду со значениями 0 и 1 в ней будут встречаться символы неопределенности «–». Например, если задана система из трех функций:

$$M_{f1}^1 = \{0\,0\,0\,0\,1\,1,\ 0\,0\,0\,1\,0\,1\},\ M_{f1}^0 = \{0\,1\,1\,0\,0\,0,\ 1\,1\,0\,0\,1\,0\};$$

$$M_{f1}^- = \{1\,0\,0\,1\,0\,1,\ 1\,0\,1\,1\,0\,1,\ 1\,1\,1\,0\,1\,0\};$$

$$M_{f2}^1 = \{0\,0\,0\ 0\,1\,1,\ 0\,1\,1\,0\,0\,0,\ 1\,0\,1\,0\,1\,0,\ 1\,1\,1\,0\,1\,0\},\ M_{f2}^0 = \{1\,0\,0\,1\,0\,1\};$$

$$M_{f2}^- = \{0\,0\,0\,1\,0\,1,\ 1\,0\,1\,1\,0\,1\};$$

$$M_{f3}^{1} = \{0\,1\,1\,0\,0\,0,\ 1\,1\,1\,0\,1\,0\},\ M_{f3}^{0} = \{0\,0\,0\,1\,0\,1,\ 1\,0\,0\,1\,0\,1,\ 1\,0\,1\,1\,0\,1\};$$
$$M_{f3}^{-} = \{0\,0\,0\,0\,1\,1,\ 1\,1\,0\,0\,1\,0\},$$

то в матричном виде она представляется булевой и троичной матрицами:

$$\begin{array}{c} \\ \boldsymbol{X} = \end{array}\begin{array}{cccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \end{array} \qquad \begin{array}{c} \\ \boldsymbol{Y} = \end{array}\begin{array}{ccc} y_1 & y_2 & y_3 \\ 1 & 1 & - \\ 1 & - & 0 \\ 0 & 1 & 1 \\ - & 0 & 0 \\ - & - & 0 \\ 0 & 1 & - \\ - & 1 & 1 \end{array}$$

Компактность задания системы булевых функций в некоторых ситуациях может быть повышена, если использовать не булеву, а троичную матрицу $\boldsymbol{X}$ (матрица $\boldsymbol{Y}$ остается булевой**)**. В этом случае значения булевых функций рассматриваемой системы будут задаваться уже не на отдельных элементах, а на интервалах булева пространства переменных $x_1, x_2, ..., x_n$, и эти интервалы будут представлены вектор-строками матрицы $\boldsymbol{X}$.

Задания

1. Привести код Грея длиной 2, 3, 4, 5.

2. Изобразить одномерный, двумерный, трехмерный и четырехмерный кубы путем последовательного удвоения их размерностей.

3. Показать на четырехмерном кубе все вершины, соответствующие элементам булева пространства, соседним с $0\,0\,0\,0$, $1\,1\,0\,0$ и $1\,0\,0\,1$.

4. Изобразить одномерную, двумерную, трехмерную, четырехмерную и пятимерную карты Карно путем последовательного наращивания их размерностей.

5. Показать на карте Карно все клетки, соответствующие элементам булева пространства, соседним с $0\,0\,0\,0$, $1\,1\,0\,0$ и $1\,0\,0\,1$.

6. Задать следующие полностью булевы определенные функции в векторной форме на гиперкубе и карте Карно:

$$f(x, y, z) = \bar{x}\,\bar{y}\,\bar{z} \vee \bar{x}\,\bar{y}z \vee x\,\bar{y}\,\bar{z} \vee xyz;$$
$$g(x, y, z) = \bar{x}y\,\bar{z} \vee \bar{x}yz \vee xy\,\bar{z} \vee xyz;$$
$$h(x, y, z) = \bar{x}yz \vee \bar{x}\,\bar{y}z \vee x\,\bar{y}\,\bar{z} \vee xy\,\bar{z};$$

$$e(x,y,z) = \bar{x}y\bar{z} \vee \bar{x}\bar{y}\bar{z} \vee \bar{x}y\bar{z} \vee xy\bar{z} \vee xy\bar{z}.$$

7. Задать следующие полностью определенные булевы функции в векторном представлении на гиперкубе и карте Карно:

$$\boldsymbol{f}(x,y,z) = 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0;$$
$$\boldsymbol{g}(x,y,z) = 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1;$$
$$\boldsymbol{h}(x,y,z) = 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 1;$$
$$\boldsymbol{e}(x,y,z) = 1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1.$$

8. Задать следующие частично определенные булевы функции в векторном представлении на гиперкубе и карте Карно:

$$\boldsymbol{f}(x,y,z) = 1 - 0\ 0\ 1\ 1 - 0;$$
$$\boldsymbol{g}(x,y,z) = 0\ 1\ 0 - 1 - 0 -;$$
$$\boldsymbol{h}(x,y,z) = 0\ 1\ 0 - 0\ 1 - 1\ 0 - 0\ 1\ 0\ 0\ 0\ 1;$$
$$\boldsymbol{e}(x,y,z) = 1 - 0\ 0 - 1\ 1 - 0\ 1\ 1\ 0 - 0\ 0\ 1.$$

9. Определить на карте Карно, образуют ли следующие множества булевых векторов интервал и если да, то показать соответствующий интервал; если нет, то привести множество интервалов, покрывающих элементы этого множества

$$A = \{1\ 1\ 0\ 0,\ 1\ 0\ 0\ 0,\ 1\ 0\ 1\ 0,\ 1\ 1\ 1\ 0\};$$
$$B = \{1\ 1\ 0\ 0\ 1,\ 1\ 0\ 0\ 0\ 1,\ 1\ 0\ 0\ 0\ 0,\ 1\ 1\ 0\ 0\ 0,\ 0\ 1\ 0\ 0\ 0,\ 0\ 0\ 0\ 0\ 1,\ 0\ 1\ 0\ 0\ 1,\ 0\ 0\ 0\ 0\ 0\};$$
$$C = \{1\ 1\ 0\ 0\ 1,\ 1\ 0\ 1\ 0\ 1,\ 1\ 0\ 0\ 0\ 0,\ 1\ 1\ 0\ 0\ 0,\ 0\ 1\ 0\ 0\ 0,\ 0\ 0\ 0\ 1\ 1,\ 0\ 1\ 0\ 0\ 1,\ 0\ 0\ 0\ 0\ 0\}.$$

10. Задать следующие булевы функции на картах Карно и определить по ним, в каком отношении (эквивалентности, реализации) находятся следующие пары булевых функций $f(x,y,z)$ и $g(x,y,z)$:

$$f(x,y,z) = \bar{x}\bar{y}\bar{z} \vee \bar{x}\bar{y}z \vee x\bar{y}\bar{z} \vee x\bar{z} \text{ и } g(x,y,z) = \bar{x}\bar{y} \vee \bar{y}\bar{z} \vee xy\bar{z} \vee x\bar{y}\bar{z};$$
$$\boldsymbol{f}(x,y,z) = 1 - 0 - 1\ 1 - 0 \text{ и } \boldsymbol{g}(x,y,z) = 0\ 1\ 0 - 1\ 1 - 0.$$

x	y	z	$f(x,y,z)$	$g(x,y,z)$
0	0	0	–	0
0	0	1	0	0
0	1	0	1	1
0	1	1	–	1
1	0	0	1	–
1	0	1	–	0
1	1	0	0	–
1	1	1	0	0

11. Привести карту Карно для мажоритарной функции от трех аргументов, которая принимает значение 1 на наборах, в которых число единичных компонент больше числа нулевых.

12. Определить, в каких отношениях (равенства, больше, меньше) находятся булевы векторы следующих пар:

0 0 0 0 0 и 0 0 0 0 1;
1 1 0 0 0 0 и 0 1 0 0 1 0;
1 1 0 0 1 1 и 0 1 0 0 1 0;
0 1 0 0 1 1 и 0 1 1 0 1 1.

13. Определить, в каких отношениях (равенства, ортогональности, пересечения, поглощения, соседства, смежности) находятся троичные векторы (и соответствующие им интервалы) следующих пар:

1 – 0 0 1 1 – 0 и 1 – 0 0 1 1 – 0;
1 – 0 0 1 1 – 1 и 1 – 0 0 1 1 – 0;
1 – 0 0 – 1 – 1 и 1 – 0 0 1 1 – 0;
1 – 0 0 – 1 – 1 и 1 0 – 0 1 1 – 1;
1 1 0 0 – 1 – 1 и 1 0 – 0 1 1 – 1;
1 1 0 0 – 1 – 1 и 1 – 1 – 1 0 – 1;
1 – – 0 – 1 – 1 и 1 – 1 – 1 – – 1.

14. Определить, образуют ли следующие множества булевых векторов интервал и если да, то привести соответствующий интервалу троичный вектор:

A = {1 1 0 0, 1 0 0 0, 1 0 1 0, 1 1 1 0};

B = {1 1 0 0 1, 1 0 0 0 1, 1 0 0 0 0, 1 1 0 0 0, 0 1 0 0 0, 0 0 0 0 1, 0 1 0 0 1, 0 0 0 0 0};

C = {1 1 0 0 1, 1 0 1 0 1, 1 0 0 0 0, 1 1 0 0 0, 0 1 0 0 0, 0 0 0 1 1, 0 1 0 0 1, 0 0 0 0 0}.

15. Для следующей пары троичных векторов

$\boldsymbol{p}$ = 1 – – 0 1 1 – 0 и $\boldsymbol{q}$ = 1 – 0 – 0 1 – 0

найти:

– дизъюнкцию $\boldsymbol{p} \vee \boldsymbol{q}$;
– конъюнкцию $\boldsymbol{p} \wedge \boldsymbol{q}$;
– дизъюнкцию с исключением $\boldsymbol{p} \oplus \boldsymbol{q}$;
– эквиваленцию $\boldsymbol{p} \sim \boldsymbol{q}$;
– импликацию $\boldsymbol{p} \rightarrow \boldsymbol{q}$.

16. Для следующей пары частично определенных булевых функций, заданных в векторном виде

$$\boldsymbol{f}(x, y, z) = 1-0011-0 \text{ и } \boldsymbol{g}(x, y, z) = 010-1-0-,$$

найти и задать в табличной и матричной формах функции

$$\boldsymbol{f} \vee \boldsymbol{g};\ \boldsymbol{f} \wedge \boldsymbol{g};\ \boldsymbol{f} \oplus \boldsymbol{g};\ \boldsymbol{f} \to \boldsymbol{g};\ \ \boldsymbol{g} \to \boldsymbol{f}.$$

17. Найти функции, двойственные данным функциям, используя принцип двойственности

$$f_1 = x\,y \oplus (y \vee z);$$
$$f_2 = (x \vee y) \oplus (x \mid y);$$
$$f_3 = (x \sim z) \vee (x \oplus y);$$
$$f_4 = x \vee y \oplus z;$$
$$f_5 = (x \to y) \oplus z;$$
$$f_6 = (z \to x) \to y;$$
$$f_7 = (x \oplus y) \to (z \to x);$$
$$f_8 = ((x \sim y) \vee z) \to x;$$
$$f_9 = (x \to y) \to (z \to x);$$
$$f_{10} = (x \mid y) \to (x \downarrow z).$$

18. Проверить на монотонность, линейность и самодвойственность:

– основные элементарные функции алгебры логики;

– мажоритарную функцию от трех аргументов;

– функции из задания 17.

19. Найти коэффициенты дизъюнктивного и конъюнктивного разложения:

– функции $f(x,y,z)= \bar{x}\,\bar{y}\,\bar{z} \vee \bar{x}\,\bar{y}\,z \vee x\,\bar{y}\,\bar{z} \vee x\,\bar{z}$ по переменной y и по переменным x и z;

– функции $g(x,y,z)= \bar{x}\,\bar{y} \vee \bar{y}\,\bar{z} \vee x\,y\,\bar{z} \vee x\,\bar{y}\,\bar{z}$ по переменной x и по переменным y и z.

7 РЕАЛИЗАЦИЯ БУЛЕВЫХ ФУНКЦИЙ КОМБИНАЦИОННЫМИ СХЕМАМИ

7.1 Функциональная полнота

Исходной информацией для синтеза логических схем является описание элементного базиса, т. е. набора элементов, из которых можно строить схему, и функциональное описание схемы в виде системы булевых функций, которые необходимо реализовать в этом базисе.

Элементом является некоторое простейшее в каком-то смысле техническое устройство. Двоичные физические сигналы на его входах и выходах связаны некоторой функциональной зависимостью, которую можно выразить функцией и считать, что элемент реализует данную булеву функцию (или систему).

Задача синтеза состоит в том, чтобы реализовать функции исходной системы в заданном элементном базисе, т. е. представить их в виде суперпозиций элементарных булевых функций, реализуемых элементами этого базиса. В качестве дополнительного требования может выступать минимум сложности (стоимости) получаемой схемы.

Одно из основных требований, предъявляемых к набору используемых логических элементов, состоит в том, чтобы из элементов этого набора можно было построить логическую схему, реализующую любую, сколь угодно сложную булеву функцию. Техническая задача отыскания такого набора элементов сводится к математической задаче определения такого набора логических функций, из которых путем суперпозиции можно получить любую функцию. Такая система логических функций называется *функционально полной*.

Функциональная полнота системы булевых функций обеспечивает возможность построения с помощью элементов соответствующего набора логических схем, реализующих любые булевы функции. Соответственно техническая задача синтеза логических схем полностью эквивалентна математической задаче представления заданных булевых функций через функции выбранной функционально полной системы.

7.1.1 Функционально полные системы функций

Система булевых функций называется *функционально полной*, если любая булева функция может быть выражена в виде суперпозиции этих функций, т. е. если множество функций $F = \{f_1, f_2, \ldots, f_p\}$ образует функционально полную систему, то любая функция h представима в виде формулы над F:

$$h = H[F] = H[f_1, f_2, \ldots, f_p].$$

Здесь запись $H[f_1, f_2, \ldots, f_p]$ не означает, что обязательно все функции из F входят в формулу $H[F]$.

Тривиальный пример функционально полной системы – множество всех булевых функций. Очевидна также полнота булева базиса $\{\neg, \vee, \wedge\}$, так как любую булеву функцию можно представить в одной из двух нормальных форм (ДНФ или КНФ). Отсюда функционально полная система из трех функций $\{\bar{x}, x \vee y, x \wedge y\}$.

Вопрос о полноте одних систем функций можно свести к вопросу о полноте других систем. Справедливо следующее утверждение.

Утверждение 7.1. Если все функции функционально полной системы представимы в виде формул через функции другой системы, то последняя система также функционально полна.

Из утверждения следует, что для того чтобы убедиться в том, что некоторая система функций G полна, достаточно выразить через ее функции любую функцию из некоторой системы F, функциональная полнота которой известна. Таким образом, можно установить функциональную полноту следующих систем.

1. Система функций $\{\bar{x}, x \vee y\}$ полна, так как функционально полна система $\{\bar{x}, x \vee y, x \wedge y\}$ и $x \wedge y = \overline{\bar{x} \vee \bar{y}}$.

2. Система функций $\{\bar{x}, x \wedge y\}$ полна, поскольку $x \vee y = \overline{\bar{x} \wedge \bar{y}}$.

3. Система функций $\{x \to y, 0\}$ полна, так как полна система $\{\bar{x}, x \vee y\}$ и $\bar{x} = x \to 0$, $x \vee y = \bar{x} \to y$.

4. Система функций $\{1, x \wedge y, x \oplus y\}$ функционально полна, так как полна система $\{\bar{x}, x \wedge y\}$ и $\bar{x} = x \oplus 1$.

С полнотой последней системы связана *алгебра Жегалкина*. В этой алгебре любая булева функция представима полиномом Жегалкина (см. пункт 6.3.4).

Существуют функционально полные системы, состоящие из одной функции.

5. Система, состоящая из функции Шеффера $\{x \mid y\}$ ($x \mid y = \overline{x \wedge y}$), полна, так как полна система $\{\bar{x}, x \wedge y\}$ и

$$\bar{x} = x \mid x, \quad x \wedge y = \overline{\overline{x \wedge y}} = \overline{\overline{x \wedge y} \wedge \overline{x \wedge y}} = (x \mid y) \mid (x \mid y).$$

6. Система, состоящая из функции Пирса $\{x \uparrow y\}$ ($x \uparrow y = \overline{x \vee y}$), функционально полна, так как полна система $\{\bar{x}, x \vee y\}$ и

$$\bar{x} = x \uparrow x, \quad x \vee y = \overline{\overline{x \vee y}} = \overline{\overline{x \vee y} \vee \overline{x \vee y}} = (x \uparrow y) \uparrow (x \uparrow y).$$

Из приведенных примеров видно, что существует целый ряд функционально полных систем. Каждая из них может быть принята за базовую для представления множества булевых функций.

Вышеприведенное утверждение позволяет доказать полноту некоторой системы булевых функций. Гораздо труднее доказать неполноту системы функций. Для доказательства неполноты системы функций можно найти такое свойство, которым обладают все ее функции и которое сохраняется при их суперпозиции. Тогда функцию, не обладающую этим свойством, нельзя будет представить в виде формулы через функции анализируемой системы.

При исследовании сохраняемых при суперпозиции свойств функций используются понятия замыкания и замкнутого класса.

7.1.2 Важнейшие замкнутые классы

Замыканием множества F булевых функций называется множество всех булевых функций, представимых в виде формул через функции множества F. Замыкание множества F обозначается через $[F]$. Отметим некоторые очевидные свойства замыкания:

1) $[F] \supseteq F$;
2) $[[F]] = [F]$;
3) если $F_1 \subseteq F_2$, то $[F_1] \subseteq [F_2]$;
4) $[F_1 \cup F_2] \supseteq [F_1] \cup [F_2]$.

Множество F функций называется *замкнутым,* если $[F] = F$.

Примером замкнутого класса служит множество всех дизъюнкций.

Рассмотрим следующие замкнутые классы булевых функций.

1. *Класс монотонных функций* ($K_м$). Напомним (см. пункт 6.3.3), что функция $f(x_1, x_2, \ldots, x_n)$ называется *монотонной*, если для любых двух наборов $\boldsymbol{\sigma}$ и $\boldsymbol{\tau}$ значений аргументов, таких, что $\boldsymbol{\sigma} \le \boldsymbol{\tau}$, имеет место неравенство $f(\boldsymbol{\sigma}) \le f(\boldsymbol{\tau})$.

Монотонными функциями являются x, $x \wedge y$, $x \vee y$, а также константы 1 и 0. Немонотонны, например, $\bar{x}, x \wedge \bar{y}, x \oplus y$.

Для доказательства замкнутости класса $K_м$ достаточно показать, что функция

$$F(x_1, x_2, \ldots, x_n) = f(f_1(x_1, x_2, \ldots, x_n), f_2(x_1, x_2, \ldots, x_n), \ldots, f_m(x_1, x_2, \ldots, x_n))$$

является монотонной, если функции $f, f_1, f_2, \ldots, f_m$ монотонны.

Пусть $\boldsymbol{\sigma}$ и $\boldsymbol{\tau}$ – два набора значений переменных $x_1, x_2, \ldots, x_n$, причем $\boldsymbol{\sigma} \leq \boldsymbol{\tau}$. В силу монотонности функций $f_1, f_2, \ldots, f_m$ имеем

$$f_1(\boldsymbol{\sigma}) \leq f_1(\boldsymbol{\tau}),\ f_2(\boldsymbol{\sigma}) \leq f_2(\boldsymbol{\tau}),\ \ldots,\ f_m(\boldsymbol{\sigma}) \leq f_m(\boldsymbol{\tau}).$$

Следовательно, для наборов значений функций имеет место неравенство

$$(f_1(\boldsymbol{\sigma}), f_2(\boldsymbol{\sigma}), \ldots, f_m(\boldsymbol{\sigma})) \leq (f_1(\boldsymbol{\tau}), f_2(\boldsymbol{\tau}), \ldots, f_m(\boldsymbol{\tau})).$$

В силу монотонности функции f имеем

$$f(f_1(\boldsymbol{\sigma}), f_2(\boldsymbol{\sigma}), \ldots, f_m(\boldsymbol{\sigma})) \leq f(f_1(\boldsymbol{\tau}), f_2(\boldsymbol{\tau}), \ldots, f_m(\boldsymbol{\tau})),$$

откуда $F(\boldsymbol{\sigma}) \leq F(\boldsymbol{\tau})$.

2. *Класс линейных функций* ($K_л$). Функция $f(x_1, x_2, \ldots, x_n)$ называется *линейной,* если ее можно представить линейным полиномом в алгебре Жегалкина:

$$f(x_1, x_2, \ldots, x_n) = a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus \ldots \oplus a_nx_n,$$

где каждый из коэффициентов a_i равен 0 или 1.

Линейными функциями являются x, $\bar{x}$, $x \oplus y$ и константы 0 и 1. Нелинейными являются $x \wedge y$, $x \vee y$.

Класс линейных функций является замкнутым, так как подстановка формул типа $a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus \ldots \oplus a_nx_n$ в такую же формулу снова дает формулу такого же типа.

3. *Класс самодвойственных функций* ($K_с$). Функция f называется *самодвойственной*, если она равна двойственной себе: $f = f^*$. Самодвойственная функция на каждой паре противоположных наборов принимает противоположные значения:

$$f(x_1, x_2, \ldots, x_n) = \bar{f}(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n).$$

Самодвойственными функциями являются функции x и $\bar{x}$, несамодвойственны, например, константы 0 и 1, а также $x \wedge y$, $x \vee y$, $x \oplus y$, $x \mid y$.

Для доказательства замкнутости класса $K_с$ достаточно показать, что функция $F = f(f_1, f_2, \ldots, f_m)$ является самодвойственной, если функции $f, f_1, f_2, \ldots, f_m$ самодвойственны, а это устанавливается непосредственно:

$$F^* = f^*(f_1^*, f_2^*, \ldots, f_m^*) = f(f_1, f_2, \ldots, f_m) = F.$$

4. *Класс функций, сохраняющих константу* 0 (K_0). Функция f сохраняет константу 0, если она принимает значение 0 на нулевых наборах:

$$f(0, 0, ..., 0) = 0.$$

К функциям, сохраняющим константу 0, относятся константа 0, а также x, $x \wedge y$, $x \vee y$, $x \oplus y$, но не относится, например, константа 1 и $x \mid y$.

Для доказательства замкнутости класса K_0 достаточно показать, что функция $f = f(f_1, f_2, \dots, f_m)$ принадлежит классу K_0, если функции $f, f_1, f_2, \dots, f_m$ принадлежат K_0. Это вытекает из цепочки равенств

$$F(\mathbf{0}) = f(f_1(0, 0,\dots, 0), f_2(0, 0,\dots, 0), \dots, f_n(0, 0,\dots, 0)) = f(0, 0,\dots, 0) = 0.$$

5. *Класс функций, сохраняющих константу* 1 (K_1). Функция f сохраняет константу 1, если она принимает значение 1 на единичных наборах:

$$f(1, 1, ..., 1) = 1.$$

К функциям, сохраняющим константу 1, относятся, например, константа 1, а также x, $x \wedge y$, $x \vee y$, но не относятся константа 0 и функции $x \oplus y$, $x \mid y$.

Замкнутость класса K_1 вытекает из цепочки равенств

$$F(\mathbf{1}) = f(f_1(1, 1,\dots, 1), f_2(1, 1,\dots, 1), \dots, f_n(1, 1,\dots, 1)) = f(1, 1,\dots, 1) = 1.$$

В таблице 7.1 показано, каким из введенных классов не принадлежат некоторые функции алгебры логики. Если функция не принадлежит некоторому классу, то соответствующая клетка таблицы помечается символом «×».

Таблица 7.1 – Непринадлежность логических функций замкнутым классам

Функция	$K_м$	$K_л$	$K_с$	K_0	K_1
0			×		×
1			×	×	
x					
$\bar{x}$	×			×	×
xy		×	×		
$x \vee y$		×	×		
$x \to y$	×	×	×	×	
$x \oplus y$	×		×		×
$x \sim y$	×		×	×	
$x \mid y$	×	×	×	×	×
$x \uparrow y$	×	×	×	×	×
$\neg(x \to y)$	×	×	×		×

7.1.3 Теорема о функциональной полноте

Ответ на вопрос о функциональной полноте системы булевых функций может быть получен с помощью теоремы Поста, дающей необходимые и достаточные условия полноты заданной системы функций.

Теорема Поста. Для того чтобы система функций была функционально полной, необходимо и достаточно, чтобы она содержала по крайней мере одну немонотонную, одну нелинейную, одну несамодвойственную, одну не сохраняющую 0 и одну, не сохраняющую 1, функции.

Необходимость следует из замкнутости всех пяти упомянутых классов.

Для доказательства достаточности нужно показать, что через функции выделенных классов можно получить функции некоторой функционально полной системы функций. Например, $\{\ \bar{x}, x \wedge y\ \}$ или $\{\ \bar{x}, x \vee y\ \}$. Возможность получения таких функций следует из следующих трех утверждений.

Утверждение 7.2. Если функция $f(x_1, x_2, \ldots, x_n)$ немонотонна, то путем подстановки констант 0 и 1 и функции x из нее можно получить функцию $\bar{x}$.

Утверждение 7.3. Если функция $f(x_1, x_2, \ldots, x_n)$ нелинейна, то путем подстановки вместо ее аргументов констант 0 и 1 и функций x и $\bar{x}$ из нее можно получить конъюнкцию.

Утверждение 7.4. Константы 0 и 1 могут быть получены из функций $f_1(x_1, x_2, \ldots, x_n) \notin K_0$, $f_2(x_1, x_2, \ldots, x_n) \notin K_1$, $f_3(x_1, x_2, \ldots, x_n) \notin K_C$.

Утверждения 7.2 и 7.3 позволяют сделать вывод о том, что система, включающая немонотонную и нелинейную функции, является функционально полной при наличии констант. Это еще не функциональная полнота в обычном смысле, так как константы с самого начала предполагаются данными. Однако такое предположение оправдано при синтезе логических схем, когда соответствующие сигналы могут быть доступными.

Система функций называется *функционально полной в слабом смысле,* если любая булева функция может быть выражена через функции этой системы и константы 0 и 1.

Очевидно, что из полноты системы функций следует ее слабая полнота.

Утверждение 7.5. Для того чтобы система функций была функционально полной в слабом смысле, необходимо и достаточно, чтобы она содержала хотя бы одну немонотонную и одну нелинейную функцию.

Необходимость следует из замкнутости классов монотонных и линейных функций и соответственно невозможности получения с помощью суперпозиций этих функций немонотонных или нелинейных функций. Достаточность следует из утверждений 7.2 и 7.3: из констант и немонотонной функции можно получить отрицание, из констант, отрицания и нелинейной функции – конъюнкцию и дизъюнкцию.

К примеру, система функций $\{x \oplus y, x \wedge y\}$ является функционально полной в слабом смысле.

7.1.4 Минимальная полная система функций

Функционально полная система называется еще *базисом*. Этот термин используется при синтезе логических схем.

Минимальной полной системой функций, или *минимальным базисом*, называется базис, который становится неполным при удалении из него любой функции.

Например, булев базис $\{\neg, \wedge, \vee\}$ не является минимальным базисом, так как при удалении конъюнкции или дизъюнкции он остается базисом.

Из теоремы Поста вытекает следующее *утверждение*: максимально возможное число функций в минимальном базисе равно четырем.

По теореме Поста число функций в базисе не превышает пяти, так как для функциональной полноты любой системы достаточно, чтобы она содержала только пять выделенных функций, остальные функции могут быть исключены без потери системой свойства полноты. Оказывается, что функция, не сохраняющая константу, либо немонотонна, либо несамодвойственна.

Пусть функция $f_0(x_1, x_2, \ldots, x_n)$ не сохраняет константу 0: $f(0, 0, \ldots, 0) = 1$. Если $f(1, 1, \ldots, 1) = 1$, то функция f несамодвойственна. Если же $f(1, 1, \ldots, 1) = 0$, то функция f немонотонна. Следовательно, максимальное число функций в минимальном базисе не больше четырех. Доказательством существования минимального базиса из четырех функций является базис $\{0, 1, x \wedge y, x \oplus y \oplus z\}$.

7.2 Реализация булевых функций

Исходной информацией для синтеза комбинационной схемы (схемы без элементов памяти) является описание элементного базиса, т. е. набора элементов, из которых можно строить схему, и функциональное описание схемы в виде системы булевых функций. Реализовать булеву функцию в некотором базисе значит представить ее в виде суперпозиции элементарных булевых функций, реализуемых элементами этого базиса.

В элементный базис могут входить самые разнообразные элементы, их выбор диктуется как уровнем развития технологии производства реальных дискретных элементов, так и требованиями, предъявляемыми к базису в целом со стороны проектировщика. Основными требованиями, которым должен удовлетворять базис, являются полнота и эффективность. Функциональная полнота базиса обеспечивает возможность построения с помощью его элементов логической схемы, реализующей произвольную булеву функцию. Понятие эффективности базиса менее формализовано. Эффективным является базис, позволяющий синтезировать лучшие в каком-то смысле схемы: более простые, надежные, дешевые и т. д.

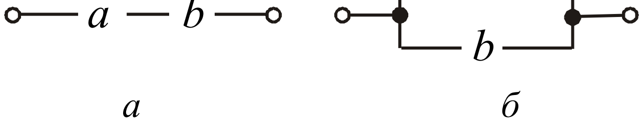
Первые электрически управляемые схемы разрабатывались на основе электромагнитных реле (в 1930-е гг.), вакуумных ламп (1940-е гг.), затем на полупроводниковых диодах и транзисторах. В 1960-е гг. была разработана интегральная схема (ИС), а затем и первые семейства ИС. Семейство ИС представляет собой набор различных ИС со сходными электрическими характеристиками, реализующих разные логические функции и имеющих различные числа входных полюсов и нагрузочные способности.

7.2.1 Релейно-контактные схемы

Первый опыт использования булевой алгебры в проектировании устройств управления связан с ее применением для описания релейных цепей. Простейшим релейным устройством является *контактный переключатель*, состоящий из пары пружинных пластин, которые могут быть либо соединены, либо разъединены, замыкая или размыкая электрическую цепь. Релейный контакт представляет собой двухполюсник, который может находиться в двух состояниях: разомкнут (его электрическая проводимость равна нулю) либо замкнут (проводимость равна 1).

В контактной схеме изображаются только контакты воспринимающих реле, так как только эти контакты замыкают и размыкают электрические цепи исполнительных элементов. Если цепь замкнута, то считается, что ее проводимость равна 1, если разомкнута, то 0. От значения проводимости зависит состояние выхода, поэтому при описании поведения контактной схемы достаточно задать ее проводимость как функцию от значений контактных переключателей.

Контактная схема с двумя выделенными полюсами (входным и выходным) называется *контактным двухполюсником*. При последовательном соединении контактных переключателей (рисунок 7.1, *а*) a и b цепь будет замкнута только в том случае, если оба переключателя замкнуты, т. е. если $a \wedge b = 1$. Параллельное соединение (рисунок 7.1, *б*) характеризуется тем, что сигнал поступает одновременно на все соединяемые параллельно переключатели. Цепь замкнута, если замкнут хотя бы один из переключателей, т. е. если $a \vee b = 1$.



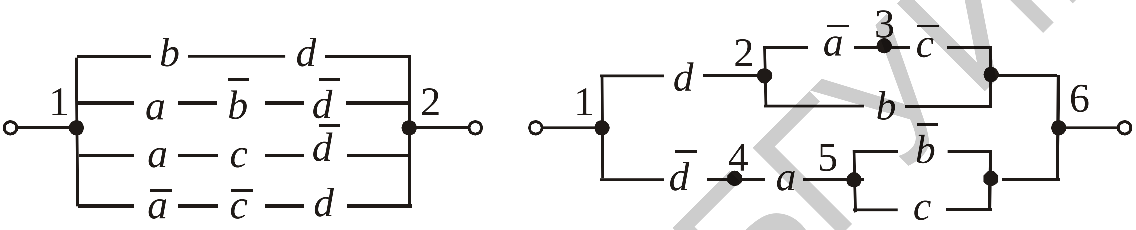
а *б*

Рисунок 7.1 – Последовательное (*а*) и параллельное (*б*) соединение контактов

Параллельные и последовательные структуры являются отправными для разработки более сложных типов контактных схем – параллельно-последовательных. Существует взаимно однозначное соответствие между контактными схемами и логическими формулами. Скобочной булевой формуле однозначно соответствует параллельно-последовательная контактная схема. Например, параллельно-последовательные контактные схемы, изображенные на рисунке 7.2, описываются следующими логическими формулами:

$$((b \cdot d) \vee (a \cdot \bar{b} \cdot \bar{d}) \vee (a \cdot c \cdot \bar{d}) \vee (\bar{a} \cdot \bar{c} \cdot d));$$

$$((d \cdot ((\bar{a} \cdot \bar{c}) \vee b)) \vee ((\bar{d} \cdot a \cdot (\bar{b} \vee c)))).$$



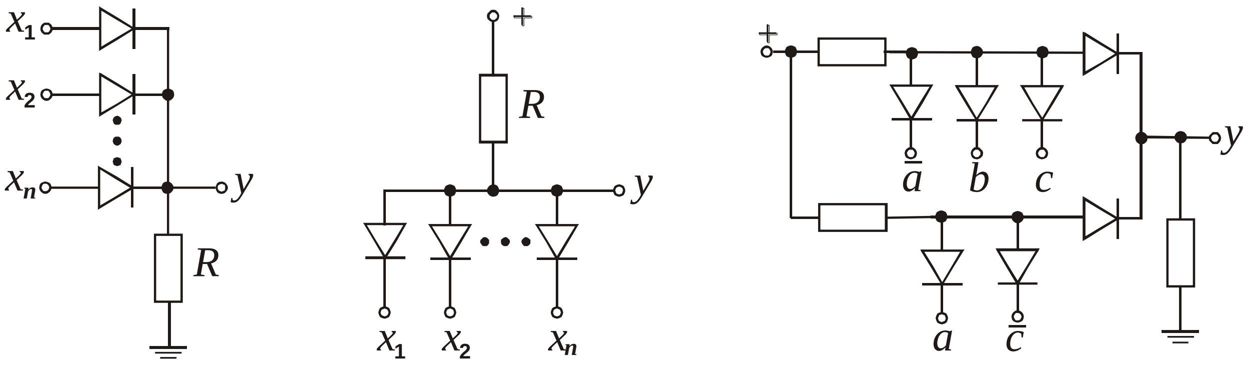
а *б*

Рисунок 7.2 – Параллельно-последовательная контактная схема, построенная по заданию функции в виде ДНФ (*а*) и скобочной формулы (*б*)

7.2.2 Схемы на транзисторах

Достаточно просто элементарные операции дизъюнкции и конъюнкции реализуются на диодах (рисунки 7.3, *а*, *б*). Диод обладает следующим свойством: в прямом направлении он проводит относительно большой ток при малом падении напряжения, поэтому его прямое сопротивление весьма мало; в обратном направлении ток мал, а падение напряжения велико, поэтому обратное сопротивление велико.

В схеме, реализующей дизъюнкцию (см. рисунок 7.3, *а*), если хотя бы на один из входов подается высокий потенциал ($x_i = 1$), через сопротивление R течет ток и выходное напряжение высокое ($y = 1$). В схеме, реализующей конъюнкцию (см. рисунок 7.3, *б*), если на все входы подается высокий потенциал (все $x_i = 1$), разности потенциалов на концах всех диодов равны нулю (диоды закрыты), ток через сопротивление R прекращается и на выходе устанавливается высокий потенциал ($y = 1$). Если хотя бы на один входи подается низкий потенциал, то через соответствующий диод и сопротивление R течет ток, в результате напряжение на выходе схемы равно нулю ($y = 0$).

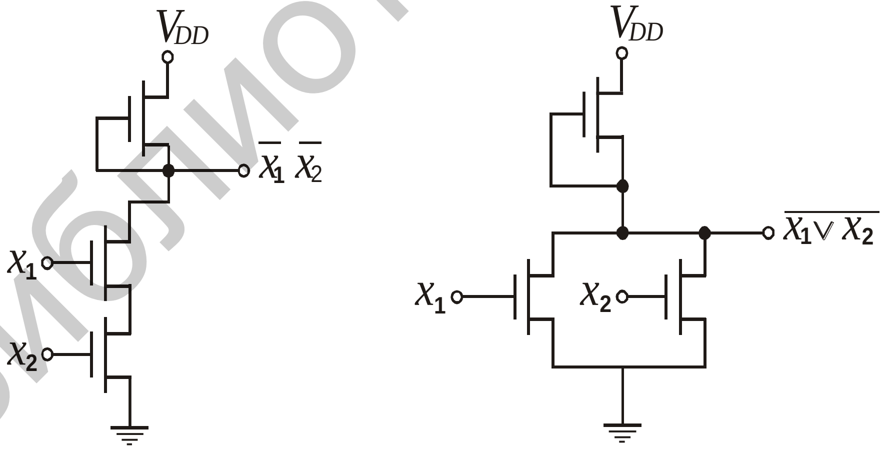


а *б* *в*

Рисунок 7.3 – Диодные схемы, реализующие дизъюнкцию (*а*), конъюнкцию (*б*) и ДНФ $y = \bar{a}\,b\,c \vee a\,\bar{c}$ (*в*)

Схемной реализации инвертора на диодах не существует, однако если допустить наличие инверсий переменных на входах диодной схемы, то ею можно реализовать ДНФ любой булевой функции (рисунок 7.3, *в*).

Инверторы и более сложные, чем на диодах, схемы можно реализовать на транзисторах. На рисунке 7.4 приведены базовые транзисторные схемы на МОП-структурах (МОП – «металл-оксид-полупроводник»). МОП-транзисторы, включенные последовательно (рисунок 7.4, *а*) и параллельно (рисунок 7.4, *б*), реализуют функции И-НЕ и ИЛИ-НЕ.



а *б*

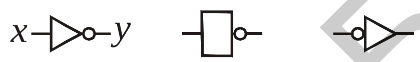
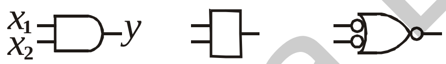
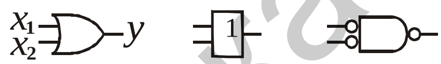
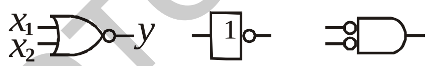
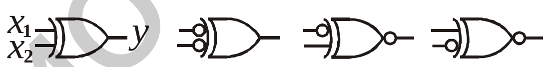
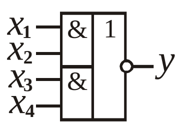
Рисунок 7.4 – МОП-структуры 2-входового И-НЕ (*а*) и 2-входового ИЛИ-НЕ (*б*)

7.2.3 Схемы из функциональных элементов

Другим типом переключательной схемы является схема из электронных логических элементов, где булевы переменные представляются сигналами в виде высокого или низкого потенциала, а элементы реализуют логические функции (например, инверсии, конъюнкции и дизъюнкции).

В наборах интегральных схем в качестве базовых логических схем используется несколько типов базовых вентилей. Их конкретная реализация, а соответственно и сложность, зависят от технологии изготовления ИС. В таблице 7.2 показаны основные типы двухвходовых вентилей с точки зрения реализуемых ими логических функций.

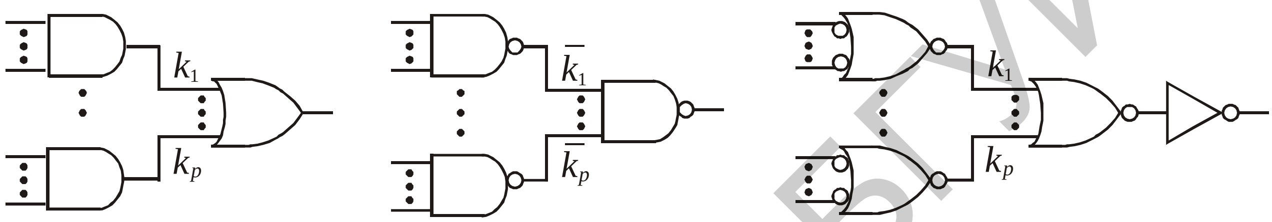
Таблица 7.2 – Основные типы логических вентилей

Название вентиля	Основные схемные обозначения	Реализуемая функция
НЕ		$y = \bar{x}$
И		$y = x_1 x_2$
ИЛИ		$y = x_1 \vee x_2$
И-НЕ		$y = \overline{x_1 x_2}$
ИЛИ-НЕ		$y = \overline{x_1 \vee x_2}$
Исключающее ИЛИ		$y = x_1 \oplus x_2$
2И-2ИЛИ-НЕ		$y = \overline{x_1 x_2 \vee x_3 x_4}$

В семействах ИС, как правило, вентили одного и того же типа (реализуемой функции) имеют разное число входов. Максимально возможное число входов вентилей в разных семействах ИС (в зависимости от технологии их изготовления) может быть разным, как правило, это 4, 6 или 8.

Наиболее разработанными являются методы синтеза схем, реализующих функции, заданные в виде ДНФ (или КНФ). Рассмотрим некоторые методы синтеза схем в базисе вентилей с произвольным и ограниченным числом входных полюсов.

1. *Реализация ДНФ в булевом базисе* (И, ИЛИ, НЕ). Отдельная ДНФ булевой функции $D = k_1 \vee k_2 \vee \ldots \vee k_p$, в которой через k_i обозначены входящие в ДНФ элементарные конъюнкции, реализуется в булевом базисе двухъярусной схемой, если не принимать во внимание возможность введения инверторов на входе и выходе (рисунок 7.5, *а*). Аналогично реализуется и система ДНФ. Элементами первого яруса служат вентили И, реализующие конъюнкции, а элементами второго – вентили ИЛИ, реализующие дизъюнкции конъюнкций ДНФ системы. При этом выходные полюсы элементов первого яруса могут соединяться с входными полюсами элементов второго яруса, а выходные полюсы элементов второго яруса служат выходными полюсами комбинационной сети в целом.



а *б* *в*

а – И / ИЛИ-реализация; *б* – И-НЕ / И-НЕ-реализация;
в – ИЛИ-НЕ / ИЛИ-НЕ-реализация

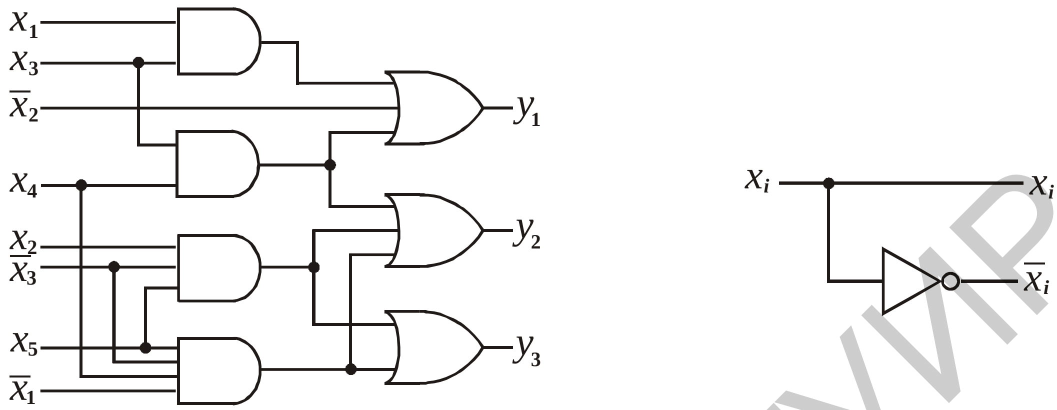
Рисунок 7.5 – Схемы, реализующие ДНФ

На рисунке 7.6 показана схема, реализующая следующую систему ДНФ:

$$y_1 = x_1 x_3 \vee x_3 x_4 \vee \bar{x}_2;$$
$$y_2 = x_3 x_4 \vee x_2 \bar{x}_3 x_5 \vee \bar{x}_1 \bar{x}_3 x_4 x_5;$$
$$y_3 = x_2 \bar{x}_3 x_5 \vee \bar{x}_1 \bar{x}_3 x_4 x_5 .$$

Заметим, что схема на рисунке 7.6, *а* изображена в предположении, что инверсии входных переменных доступны (получены вне синтезируемой схемы). В противном случае понадобилось бы введение в схему дополнительного яруса, состоящего из подсхем получения инверсий (рисунок 7.6, *б*).

Очевидно, что существует прямая связь между сложностью системы ДНФ и реализующей ее схемы в булевом базисе: схема содержит столько элементов ИЛИ, сколько ДНФ имеет система, и столько элементов И, сколько различных конъюнкций входит во все ДНФ. Суммарное число входных полюсов всех элементов схемы равно сумме рангов всех конъюнкций и дизъюнкций системы ДНФ. Последнее число особенно важно, так как площадь кристалла ИС, необходимая для реализации вентиля, пропорциональна числу его входных полюсов. Оптимизация схемной реализации булевой функции в булевом базисе сводится к нахождению реализующей ее кратчайшей (с наименьшим числом конъюнкций) или минимальной (с наименьшим суммарным числом литералов во всех конъюнкциях) ДНФ.



а б

а – И / ИЛИ-реализация; б – реализация инверсии переменной

Рисунок 7.6 – Реализация системы ДНФ в булевом базисе

2. *Реализация ДНФ в других базисах.* В современной микросхемотехнике широко распространены логические вентили, реализующие инверсные булевы функции типа И-НЕ и ИЛИ-НЕ, т. е. отрицания конъюнкции или дизъюнкции нескольких переменных. Функция И-НЕ принимает значение 0, если все аргументы принимают значение 1, и обращается в 1 в противном случае. Функция ИЛИ-НЕ принимает значение 1, если все аргументы принимают значение 0, и обращается в 0 в противном случае.

Известно, что каждая из этих функций образует полную систему, а значит, любая булева функция может быть представлена в виде некоторой суперпозиции только одного типа функций И-НЕ или ИЛИ-НЕ. Это представление легко получить, если исходить из представления булевой функции в виде ДНФ $D = k_1 \vee k_2 \vee \ldots \vee k_p$, а затем, опираясь на обобщение закона де Моргана на случай многоместных операций дизъюнкции и конъюнкции, заменить эту ДНФ на равносильное выражение:

$$D = \overline{\overline{k_1 \vee k_2 \vee \ldots \vee k_p}} = \overline{\bar{k}_1 \wedge \bar{k}_2 \wedge \ldots \wedge \bar{k}_p}\,.$$

Полученное выражение представляет, как нетрудно видеть, суперпозицию операторов И-НЕ и реализуется двухъярусной схемой из этих элементов. На выходе элементов первого яруса реализуются инверсии конъюнкций k_i исходной ДНФ. Таким образом, синтез схемы в базисе И-НЕ может быть сведен к синтезу соответствующей двухъярусной схемы в булевом базисе с последующей заменой всех элементов на элементы И-НЕ (см. рисунок 7.5, *б*). Очевидно, что при синтезе схем в базисе И-НЕ можно в полной мере использовать методы минимизации булевых функций в классе ДНФ.

Аналогично синтез схем из элементов ИЛИ-НЕ также можно свести к синтезу соответствующей двухъярусной схемы в булевом базисе (см. рисунок 7.5, *в*). Представим конъюнкции k_i ДНФ в виде $k_i = b_{i1}\, b_{i2} \ldots b_{iq}$ (где b_{ij} – переменная x_{ij} или ее отрицание $\bar{x}_{ij}$), возьмем отрицание ДНФ D и применим равносильные преобразования:

$$\bar{D} = \overline{k_1 \vee k_2 \vee \ldots \vee k_p} = \overline{\bar{\bar{k}}_1 \vee \bar{\bar{k}}_2 \vee \ldots \vee \bar{\bar{k}}_p} =$$

$$= \overline{\overline{\bar{b}_{11} \vee \bar{b}_{12} \vee \ldots \vee \bar{b}_{1q_1}} \vee \overline{\bar{b}_{21} \vee \bar{b}_{22} \vee \ldots \vee \bar{b}_{2q_2}} \vee \ldots \vee \overline{\bar{b}_{p1} \vee \bar{b}_{p2} \vee \ldots \vee \bar{b}_{pq_p}}}.$$

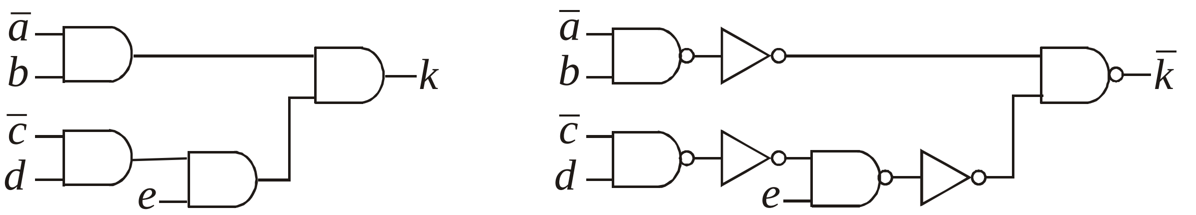
Отсюда следует, что при реализации булевой функции $f(x_1, x_2, \ldots, x_n)$ можно перейти к двойственной функции $\bar{f}(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$ и реализовать эту функцию в булевом базисе, а затем заменить все элементы полученной схемы элементами ИЛИ-НЕ.

3. *Построение схемы из реальных логических элементов*. При решении задач практического синтеза комбинационных схем приходится учитывать ряд ограничений, свойственных реальным элементам, основным из них является число k_{in} входных полюсов. В этом случае не для любой булевой функции можно найти реализующую ее двухъярусную схему в булевом базисе.

Для построения многоярусной схемы потребуется провести разложение «длинных» элементарных конъюнкций и дизъюнкций системы ДНФ. Для этого можно применить, например, факторизационный метод синтеза, манипулирующий с алгебраической формой представления полностью определенных булевых функций в виде ДНФ, КНФ и факторизованных форм.

Конъюнкции и дизъюнкции ранга, большего чем максимальное число k_{in} входов, одним элементом реализованы быть не могут, а требуют разбиения и реализации по частям. Рассматривая конъюнкцию как множество литералов, делим это множество на группы, которые будем в дальнейшем называть *факторами*.

Каждому фактору можно сопоставить в схеме элемент И для конъюнкций или ИЛИ для дизъюнкций, на входные полюсы которого подаются переменные и их инверсии, соответствующие литералам фактора. Если число факторов одной конъюнкции не больше k_{in}, то выходы схем, соответствующих этим факторам, подаются на входы элемента И, реализующего данную конъюнкцию. Если число факторов больше чем k_{in}, то аналогичные схемы вводятся для объединений факторов и т. д. (рисунок 7.7). Таким же образом реализуются и дизъюнкции большого ранга.



а *б*

а – И; *б* – И-НЕ

Рисунок 7.7 – Реализация конъюнкции $k = \bar{a}b\,\bar{c}de$ на двухвходовых элементах

7.2.4 Реализация булевых функций на программируемой логической матрице

Широкое использование при проектировании комбинационной логики (схем без памяти) получили двумерные регулярные структуры типа программируемых логических матриц (ПЛМ).

Структура ПЛМ (рисунок 7.8) представляется в виде двух программируемых транзисторных матриц И и ИЛИ с варьируемыми размерами. Каждая из матриц состоит из строк и столбцов, на пересечении которых находятся транзисторы (или схемы из транзисторов). Соединения транзисторов с шинами матричной структуры выполнены в виде плавких перемычек. На этапе реализации логической схемы на базе ПЛМ из нее удаляются некоторые транзисторы путем разрушения соответствующих связей импульсом тока или химическим травлением. Строкам такой структурной матрицы соответствуют входные и выходные сигналы, столбцам – внутренние сигналы, передаваемые из матрицы И в матрицу ИЛИ.

Входные сигналы и их инверсии поступают на горизонтальные шины матрицы И. В столбцах этой матрицы реализуются элементарные конъюнкции входных переменных, соответствующих тем строкам, на пересечении с которыми в этих столбцах находятся транзисторы. Сигналы со столбцов матрицы И поступают на входы матрицы ИЛИ. Строка матрицы ИЛИ реализует дизъюнкцию сигналов, соответствующих тем столбцам, на пересечении с которыми в строке есть транзисторы. Таким образом, ПЛМ реализует систему дизъюнктивных нормальных форм.

Площадь ПЛМ условно оценивается как $(n + m) \times k$, где n – число входных шин ПЛМ или строк матрицы И, m – число выходных шин ПЛМ или строк матрицы ИЛИ, k – число промежуточных шин ПЛМ или столбцов матриц И и ИЛИ.

Минимизация площади ПЛМ в значительной степени сводится к задаче минимизации системы булевых функций в классе ДНФ.

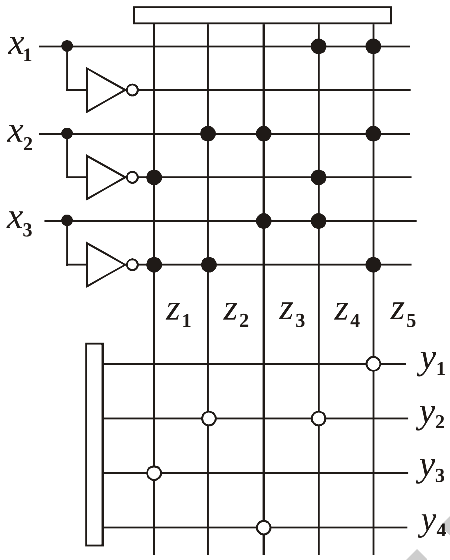


Рисунок 7.8 – Программируемая логическая матрица

В ПЛМ, приведенной на рисунке 7.8, матрица И реализует следующие элементарные конъюнкции:

$$z_1 = \bar{x}_2\,\bar{x}_3;\quad z_2 = x_2\,\bar{x}_3;\quad z_3 = x_2 x_3;\quad z_4 = x_1\,\bar{x}_2\,x_3;\quad z_5 = x_1 x_2\,\bar{x}_3.$$

На выходах ПЛМ (и в строках матрицы ИЛИ) реализуется система булевых функций:

$$y_1 = x_1 x_2\,\bar{x}_3;$$
$$y_2 = x_2\,\bar{x}_3 \vee x_1\,\bar{x}_2 x_3;$$
$$y_3 = \bar{x}_2\,\bar{x}_3;$$
$$y_4 = x_2 x_3.$$

7.3 Минимизация булевых функций

Задача минимизации булевой функции заключается в том, чтобы найти наиболее компактное ее представление в виде суперпозиции простых булевых функций, составляющих некоторую функционально полную систему. Наиболее детально эта задача исследована в классе дизъюнктивных нормальных форм – для случая функционально полной системы, состоящей из дизъюнкции, конъюнкции и отрицания.

7.3.1 Упрощение дизъюнктивных нормальных форм

Одна и та же булева функция может быть представлена различными дизъюнктивными нормальными формами (в этом случае говорят, что они эквивалентны). Например, эквивалентны следующие две ДНФ:

$$D_1 = x\,\bar{y}\,\bar{z} \vee xz\,\bar{t} \vee \bar{x}yz \vee \bar{y}zt \quad \text{и} \quad D_1 = x\,\bar{y} \vee yz\,\bar{t} \vee \bar{x}zt.$$

Возникает задача нахождения среди эквивалентных форм оптимальной в каком-то смысле, например минимальной по числу литералов. Решение такого рода задач имеет важные приложения в практике проектирования логических схем, поскольку сложность последних определяется размером ДНФ, и чем она проще, тем компактнее схема.

Получение простейших дизъюнктивных нормальных форм для заданной булевой функции представляет собой сложную задачу, известную под названием *минимизация булевой функции* (в классе ДНФ). За критерий простоты принимается общее число литералов в формуле, и в этом случае говорят о поиске *минимальной ДНФ*. В качестве другого критерия часто используется число элементарных конъюнкций в ДНФ. Дизъюнктивная нормальная форма, простейшая в этом смысле, называется *кратчайшей*. Задачи нахождения кратчайшей и минимальной ДНФ относятся к классу трудно решаемых задач.

В некоторых случаях на практике достаточно приемлемым приближением к оптимальной форме может служить *безызбыточная* ДНФ. ДНФ называется безызбыточной, если из нее нельзя удалить ни одной элементарной конъюнкции и ни одного литерала в какой-либо конъюнкции (не изменив при этом представленную булеву функцию). Часто безызбыточная ДНФ не сильно отличается от кратчайшей и минимальной ДНФ по числу элементарных конъюнкций и букв в конъюнкциях.

Ниже рассматриваются некоторые методы минимизации ДНФ.

7.3.2 Локальные методы упрощения ДНФ

Простейшие методы упрощения дизъюнктивных нормальных форм имеют локальный характер – путем равносильной замены упрощаются отдельные части ДНФ, ограниченные по числу входящих в них конъюнктивных термов. Например, при рассмотрении пар конъюнкций используют следующие три операции: *поглощение, склеивание* и *удаление литерала*.

1. *Поглощение* определяется формулой $x \vee xy = x$.

Говорят, что элементарная конъюнкция xy поглощается переменной x. Данная равносильность вытекает из тождественных преобразований:

$$x \vee xy = x(1 \vee y) = x.$$

Формула обобщается также и на случай элементарных конъюнкций:

$$k_i \vee k_i k_j = k_i.$$

2. *Склеивание* определяется формулой $xy \vee \bar{x}y = y$.

Говорят, что две элементарные конъюнкции xy и $\bar{x}y$ склеиваются по переменной x. Эта равносильность вытекает из тождественных преобразований

$$xy \vee \bar{x}y = y(x \vee \bar{x}) = y.$$

Формула обобщается на случай элементарных конъюнкций:

$$xk_i \vee \bar{x}k_i = k_i.$$

Склеиваемые элементарные конъюнкции xk_i и $\bar{x}k_i$ являются соседними по единственной переменной x.

3. *Удаление литерала* производится по формуле $x \vee \bar{x}y = x \vee y$.

Эта равносильность вытекает из тождественных преобразований

$$x \vee \bar{x}y = x(y \vee \bar{y}) \vee \bar{x}y = (xy \vee x\bar{y}) \vee \bar{x}y =$$

$$= (xy \vee x\bar{y}) \vee (xy \vee \bar{x}y) = x(y \vee \bar{y}) \vee y(x \vee \bar{x}) = x \vee y.$$

Формула обобщается на случай элементарных конъюнкций:

$$x \vee \bar{x}k_i = x \vee k_i.$$

При рассмотрении троек конъюнкций может оказаться полезной операция *обобщенного склеивания*, применяемая к смежным конъюнкциям (в которых присутствует ровно одна переменная, входящая в одну из них под знаком отрицания, а в другую – без этого знака). Данная операция определяется формулой

$$xk_i \vee \bar{x}k_j \vee k_i k_j = xk_i \vee \bar{x}k_j.$$

Заметим, что обратная операция (замена правой части равносильности на левую, т. е. добавление в формулу терма $k_i k_j$) известна под названием *резолюция*, при этом добавляемый терм $k_i k_j$ называется *резольвентой*.

Продемонстрируем приемы локального упрощения ДНФ на следующем примере:

$$\bar{x}\,\bar{y}z \vee \bar{x}y\bar{z} \vee \bar{x}yz \vee x\bar{y}z \vee xyz =$$

$$= (\bar{x}\,\bar{y}z \vee \bar{x}yz) \vee (x\bar{y}z \vee xyz) \vee \bar{x}y\bar{z} =$$

$$= \bar{x}z \vee xz \vee \bar{x}y\bar{z} = (\bar{x}z \vee xz) \vee \bar{x}y\bar{z} = z \vee \bar{x}y\bar{z} = z \vee \bar{x}y.$$

При выполнении упрощений ДНФ иногда полезным оказывается дублирование элементарных конъюнкций: $k_i = k_i \vee k_i$. Например, этот прием полезен при упрощении следующей ДНФ (здесь заводятся две дополнительные копии последней элементарной конъюнкции):

$$\bar{x} y \bar{z} \vee x \bar{y} \bar{z} \vee xyz \vee xy\bar{z} =$$

$$= (\bar{x} y \bar{z} \vee xy\bar{z}) \vee (x \bar{y} \bar{z} \vee xy\bar{z}) \vee (xyz \vee xy\bar{z}) = y\bar{z} \vee x\bar{z} \vee xy.$$

7.3.3 Сокращенные и минимальные дизъюнктивные нормальные формы

Задача минимизации заключается в поиске такой ДНФ для заданной булевой функции f, которая содержала бы минимальное число конъюнкций или литералов. При решении этой задачи существенную роль играют понятия импликанты и простой импликанты булевой функции.

Булева функция $g(x_1, x_2, ..., x_n)$ называется *импликантой* булевой функции $f(x_1, x_2,..., x_n)$, если на любом наборе значений переменных $x_1, x_2,...,x_n$, на котором значение функции g равно 1, значение функции f также равно 1. Другими словами, импликантой булевой функции $f(\boldsymbol{x})$ называется такая булева функция $g(\boldsymbol{x})$, которая имплицирует функцию f (имеется в виду формальная импликация): $g(\boldsymbol{x}) \Rightarrow f(\boldsymbol{x})$. Характеристическое множество M_g^1 импликанты $g(\boldsymbol{x})$ целиком содержится в единичной области M_f^1 функции $f(\boldsymbol{x})$: $M_g^1 \subseteq M_f^1$.

Отношение импликации транзитивно, соответственно и отношение «быть импликантой» также, т. е. если

$$g(\boldsymbol{x}) \Rightarrow h(\boldsymbol{x}) \text{ и } h(\boldsymbol{x}) \Rightarrow f(\boldsymbol{x}), \text{ то } g(\boldsymbol{x}) \Rightarrow f(\boldsymbol{x}).$$

Если при этом $g(\boldsymbol{x}) \neq h(\boldsymbol{x})$, то можно сказать, что функция $h(\boldsymbol{x})$ ближе к функции $f(\boldsymbol{x})$, чем функция $g(\boldsymbol{x})$, так как $M_g^1 \subseteq M_h^1 \subseteq M_f^1$ (рисунок 7.9).

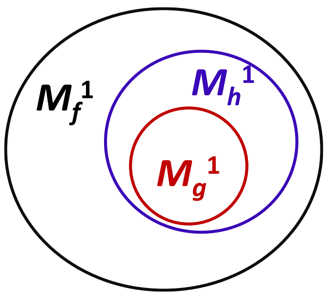


Рисунок 7.9 – Соотношение характеристических множеств функций $g(\boldsymbol{x}) \Rightarrow h(\boldsymbol{x}) \Rightarrow f(\boldsymbol{x})$

Учитывая эквивалентность отношений $g(\boldsymbol{x}) \Rightarrow f(\boldsymbol{x})$ и $M_g{}^1 \subseteq M_f{}^1$, можно подсчитать число различных импликант заданной функции $f(\boldsymbol{x})$: оно оказывается равным 2^m, где $m = |M_f{}^1|$. Одной из этих импликант оказывается сама функция $f(\boldsymbol{x})$. Например, функции $f(x_1, x_2, x_3, x_4)$, $g(x_1, x_2, x_3, x_4)$ и $h(x_1, x_2, x_3, x_4)$ с характеристическими множествами:

$M_f{}^1$:

0	0	0	1
0	0	1	0
0	1	0	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	0	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

$M_h{}^1$:

1	0	0	0
1	0	0	1
1	1	0	0
1	1	0	1

$M_g{}^1$:

1	0	0	1
1	1	0	0

(7.1)

связаны следующим образом: $g(\boldsymbol{x}) \Rightarrow h(\boldsymbol{x}) \Rightarrow f(\boldsymbol{x})$, а их характеристические множества как $M_g{}^1 \subseteq M_h{}^1 \subseteq M_f{}^1$.

Говоря об импликантах при минимизации булевых функций в классе ДНФ, обычно накладывают ограничение на тип импликанты, считая, что ею может служить только элементарная конъюнкция. При таком соглашении особый интерес представляют ближние к минимизируемой функции импликанты, т. е. такие элементарные конъюнкции, которые перестают быть импликантами при удалении из них любого литерала. Такие импликанты принято называть *простыми*.

Например, элементарные конъюнкции $x_1 x_2 \bar{x}_3 \bar{x}_4$, $x_1 \bar{x}_3 \bar{x}_4$, $x_1 \bar{x}_3$ являются импликантами функции $f(x_1, x_2, x_3, x_4)$, определенной выше, но только одна из них, $x_1 \bar{x}_3$, является простой импликантой.

Нетрудно заметить, что простые импликанты функции $f(\boldsymbol{x})$ соответствуют максимальным интервалам характеристического множества $M_f{}^1$ этой функции. При поиске простейших ДНФ типа безызбыточных, когда из ДНФ нельзя (без нарушения представляемой функции) исключить ни одной конъюнкции и ни одного литерала в ней, есть смысл ограничиться рассмотрением именно простых импликант.

Из определения формальной импликации следует, что если $g(\boldsymbol{x}) \Rightarrow f(\boldsymbol{x})$ и $h(\boldsymbol{x}) \Rightarrow f(\boldsymbol{x})$, то и $g(\boldsymbol{x}) \vee h(\boldsymbol{x}) \Rightarrow f(\boldsymbol{x})$.

Отсюда справедливы следующие утверждения.

1. Дизъюнкция любого множества импликант булевой функции также является импликантой этой функции, т. е. если $g(\boldsymbol{x})$ и $h(\boldsymbol{x})$ являются импликантами функции $f(\boldsymbol{x})$, то $g(\boldsymbol{x}) \vee h(\boldsymbol{x})$ – также импликанта функции $f(\boldsymbol{x})$.

2. Дизъюнкция всех импликант булевой функции совпадает с этой функцией.

3. Дизъюнкция всех простых импликант булевой функции совпадает с этой функцией.

Дизъюнкция всех простых импликант булевой функции называется *сокращенной* дизъюнктивной нормальной формой этой функции. Очевидно, что любая булева функция имеет единственную сокращенную ДНФ.

Для определенной выше функции $f(x_1, x_2, x_3, x_4)$ (7.1) сокращенная ДНФ имеет следующий вид:

$$f = x_1 x_2 \vee x_1 \bar{x}_3 \vee x_2 \bar{x}_4 \vee x_2 \bar{x}_3 \vee \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_3 x_4 .$$

Эта функция может быть задана следующей троичной матрицей:

$$M_f^1 = \begin{array}{cccc c} x_1 & x_2 & x_3 & x_4 & \\ 1 & 1 & – & – & 1 \\ 1 & – & 0 & – & 2 \\ – & 1 & – & 0 & 3 \\ – & 1 & 0 & – & 4 \\ 0 & – & 1 & 0 & 5 \\ – & – & 0 & 1 & 6 \end{array}$$

Сокращенная ДНФ может оказаться более экономным способом представления булевой функции, чем совершенная ДНФ. Однако в большинстве случаев она допускает дальнейшие упрощения за счет того, что некоторые из простых импликант могут поглощаться дизъюнкциями других простых импликант. Например, в приведенной выше сокращенной ДНФ простая импликанта $x_2 \bar{x}_3$ поглощается дизъюнкцией простых импликант $x_2 \bar{x}_4$ и $\bar{x}_3 x_4$, так что

$$f = x_1 x_2 \vee x_1 \bar{x}_3 \vee x_2 \bar{x}_4 \vee x_2 \bar{x}_3 \vee \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_3 x_4 =$$
$$= x_1 x_2 \vee x_1 \bar{x}_3 \vee x_2 \bar{x}_4 \vee \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_3 x_4.$$

Дизъюнкция простых импликант булевой функции называется *безызбыточной* (или *тупиковой*), если она представляет эту функцию и не содержит импликант, поглощаемых другими. Из безызбыточной ДНФ нельзя удалить ни одну элементарную конъюнкцию и ни один литерал без изменения представляемой ею булевой функции. Безызбыточная ДНФ получается из сокращенной путем исключения всех простых импликант, поглощаемых дизъюнкциями других импликант.

Нетрудно видеть, что в отличие от сокращенной ДНФ, однозначно определяемой функцией, для одной и той же функции может существовать, в общем случае, не одна безызбыточная ДНФ. Например, рассматриваемая булева функция $f(x_1, x_2, x_3, x_4)$ (7.1) имеет две безызбыточные ДНФ:

$$f = x_1 x_2 \vee x_1 \bar{x}_3 \vee x_2 \bar{x}_4 \vee \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_3 x_4; \tag{7.2}$$
$$f = x_1 x_2 \vee x_1 \bar{x}_3 \vee x_2 \bar{x}_3 \vee \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_3 x_4 .$$

ДНФ, содержащая наименьшее число импликант, называется *кратчайшей*. ДНФ, содержащая наименьшее число литералов, называется *минимальной*.

Справедливы следующие утверждения.

1. Любая минимальная ДНФ булевой функции представляет собой дизъюнкцию некоторых простых импликант этой функции и является безызбыточной ДНФ.

2. Булева функция может иметь несколько различных кратчайших и минимальных ДНФ.

3. Существуют безызбыточные ДНФ, не являющиеся ни кратчайшими, ни минимальными.

4. Существуют кратчайшие ДНФ, не являющиеся минимальными.

5. Существуют кратчайшие ДНФ, которые не являются безызбыточными: не все члены этих ДНФ простые импликанты.

К примеру, найденные выше безызбыточные ДНФ функции (7.2) являются кратчайшими и минимальными.

Первое утверждение лежит в основе двухэтапных методов решения задачи минимизации булевых функций. Эти методы состоят из следующих двух основных этапов:

– находятся все простые импликанты заданной булевой функции, т. е. ее сокращенная ДНФ;

– отыскиваются подмножества простых импликант, реализующие минимизируемую функцию, т. е. строятся безызбыточные ДНФ, из которых выбирается минимальная ДНФ.

7.3.4 Получение множества всех простых импликант

Описываемый ниже метод нахождения множества всех простых импликант был предложен первоначально Квайном, а затем усовершенствован Мак-Класки. Метод предполагает, что функция f задана первоначально в виде совершенной ДНФ (составленной из полных элементарных конъюнкций).

Метод Квайна основан на применении двух операций, называемых операциями *неполного склеивания* и *поглощения*:

$$x\,k \vee \bar{x}\,k \;=\; k \vee x\,k \;\vee\; \bar{x}\,k;$$

$$x \vee \; x\,k \;=\; x,$$

выполняемых над парами членов преобразуемой ДНФ до тех пор, пока она не превратится из совершенной ДНФ в сокращенную, представляющую собой дизъюнкцию всех простых импликант.

Теорема Квайна. Если в совершенной ДНФ булевой функции выполнить все операции неполного склеивания, а затем все операции поглощения, то получится сокращенная ДНФ этой функции.

Необходимость использования именно неполного склеивания связана с тем, что один и тот же член ДНФ может склеиваться с несколькими другими, образуя при этом различные импликанты. Поэтому при проведении операций склеивания каждый член следует оставлять в выражении для возможного использования его при других склеиваниях.

Полученная после проведения всех операций неполного склеивания дизъюнктивная нормальная форма будет содержать, кроме всех простых импликант, и импликанты, отличные от простых, которые отсеются в результате выполнения операций поглощения. Необходимо подчеркнуть, что исключить с помощью операции элементарного поглощения можно лишь те элементарные конъюнкции, к которым были применены все возможные склеивания. Нарушение этого условия может привести к ДНФ, которая не будет сокращенной.

Чтобы получить заведомо все простые импликанты, операции неполного склеивания и элементарного поглощения необходимо производить в определенном порядке, диктуемом следующим алгоритмом Квайна:

1. Минимизируемая булева функция f от произвольного числа n переменных записывается в совершенной ДНФ f^0.

2. Отправляясь от ДНФ f^0, строится последовательность ДНФ: $f^1, f^2, \ldots$ до тех пор, пока какие-либо две ДНФ f^p и f^{p+1} не совпадут между собой.

3. Переход от ДНФ f^i к ДНФ f^{i+1} ($n = 0, 1, 2,\ldots, p$) производится по следующему правилу: в форме f^i выполняются все операции неполного склеивания, применимые к элементарным конъюнкциям ранга $n - i$, после чего исключаются все элементарные конъюнкции ранга $n - i$, к которым применима операция элементарного поглощения.

4. Результатом f^p применения алгоритма Квайна к совершенной ДНФ f^0 является сокращенная ДНФ булевой функции f.

Заметим, что обоснованием шага 3 алгоритма является то, что склеиваться могут только элементарные конъюнкции одного ранга (с одинаковым числом литералов), и если к какой-то конъюнкции не была применима операция склеивания на i-м шаге, дальше она уже гарантированно ни с чем не склеится.

Продемонстрируем работу алгоритма Квайна на примере булевой функции, заданной следующей совершенной ДНФ:

$$f = x_1 x_2 x_3 \vee x_1 x_2 \bar{x}_3 \vee x_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 x_3 .$$

Операции неполного склеивания можно применить к первому и второму, первому и третьему, а также к первому и четвертому членам ДНФ. В результате получаем

$$f^{1'} = (x_1 x_2 x_3 \vee x_1 x_2 \bar{x}_3 \vee x_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 x_3) \vee x_1 x_2 \vee x_1 x_3 \vee x_2 x_3 .$$

После выполнения операций элементарного поглощения ДНФ преобразуется к виду

$$f^1 = x_1 x_2 \vee x_1 x_3 \vee x_2 x_3 .$$

Далее операция неполного склеивания не применима, и ДНФ f^1 задает сокращенную ДНФ функции f. Таким образом, функция f имеет три простые импликанты.

Для булевых функций от большого числа переменных применение алгоритма Квайна в описанном выше виде становится затруднительным. Затруднения возникают в связи с громоздкостью полных элементарных конъюнкций и большим числом продуктов неполного склеивания.

Продемонстрируем громоздкость вычислений алгоритма Квайна на примере приведенной выше (7.1) сравнительно простой функции f четырех аргументов. Ее совершенная ДНФ, принимаемая в качестве отправной, имеет вид

$$f^0 = \bar{x}_1 \bar{x}_2 \bar{x}_3 x_4 \vee \bar{x}_1 \bar{x}_2 x_3 \bar{x}_4 \vee \bar{x}_1 x_2 \bar{x}_3 \bar{x}_4 \vee \bar{x}_1 x_2 \bar{x}_3 x_4 \vee \bar{x}_1 x_2 x_3 \bar{x}_4 \vee x_1 \bar{x}_2 \bar{x}_3 \bar{x}_4 \vee$$
$$\vee x_1 \bar{x}_2 \bar{x}_3 x_4 \vee x_1 x_2 \bar{x}_3 \bar{x}_4 \vee x_1 x_2 \bar{x}_3 x_4 \vee x_1 x_2 x_3 \bar{x}_4 \vee x_1 x_2 x_3 x_4 .$$

После выполнения всех операций неполного склеивания получаем

$$f^{1'} = \bar{x}_1 \bar{x}_2 \bar{x}_3 x_4 \vee \bar{x}_1 \bar{x}_2 x_3 \bar{x}_4 \vee \bar{x}_1 x_2 \bar{x}_3 \bar{x}_4 \vee \bar{x}_1 x_2 \bar{x}_3 x_4 \vee \bar{x}_1 x_2 x_3 \bar{x}_4 \vee$$
$$\vee x_1 \bar{x}_2 \bar{x}_3 \bar{x}_4 \vee x_1 \bar{x}_2 \bar{x}_3 x_4 \vee x_1 x_2 \bar{x}_3 \bar{x}_4 \vee x_1 x_2 \bar{x}_3 x_4 \vee x_1 x_2 x_3 \bar{x}_4 \vee$$
$$\vee x_1 x_2 x_3 x_4 \vee \bar{x}_1 \bar{x}_3 x_4 \vee \bar{x}_2 \bar{x}_3 x_4 \vee \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_1 x_2 \bar{x}_3 \vee \bar{x}_1 x_2 \bar{x}_4 \vee$$
$$\vee x_2 \bar{x}_3 \bar{x}_4 \vee x_1 \bar{x}_2 \bar{x}_3 \vee x_1 \bar{x}_3 \bar{x}_4 \vee x_2 \bar{x}_3 x_4 \vee x_2 x_3 \bar{x}_4 \vee x_1 \bar{x}_3 x_4 \vee$$
$$\vee x_1 x_2 \bar{x}_3 \vee x_1 x_2 \bar{x}_4 \vee x_1 x_2 x_4 \vee x_1 x_2 x_3 .$$

После выполнения всех операций поглощения элементарных конъюнкций ранга 4 получаем следующую ДНФ:

$$f^1 = \bar{x}_1 \bar{x}_3 x_4 \vee \bar{x}_2 \bar{x}_3 x_4 \vee \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_1 x_2 \bar{x}_3 \vee \bar{x}_1 x_2 \bar{x}_4 \vee x_2 \bar{x}_3 \bar{x}_4 \vee x_1 \bar{x}_2 \bar{x}_3 \vee$$
$$\vee x_1 \bar{x}_3 \bar{x}_4 \vee x_2 \bar{x}_3 x_4 \vee x_2 x_3 \bar{x}_4 \vee x_1 \bar{x}_3 x_4 \vee x_1 x_2 \bar{x}_3 \vee x_1 x_2 \bar{x}_4 \vee x_1 x_2 x_4 \vee x_1 x_2 x_3.$$

После очередного цикла выполнения всех операций неполного склеивания получаем

$$f^{2'} = \bar{x}_1 \bar{x}_3 x_4 \vee \bar{x}_2 \bar{x}_3 x_4 \vee \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_1 x_2 \bar{x}_3 \vee \bar{x}_1 x_2 \bar{x}_4 \vee x_2 \bar{x}_3 \bar{x}_4 \vee x_1 \bar{x}_2 \bar{x}_3 \vee$$
$$\vee x_1 \bar{x}_3 \bar{x}_4 \vee x_2 \bar{x}_3 x_4 \vee x_2 x_3 \bar{x}_4 \vee x_1 \bar{x}_3 x_4 \vee x_1 x_2 \bar{x}_3 \vee x_1 x_2 \bar{x}_4 \vee x_1 x_2 x_4 \vee$$
$$\vee x_1 x_2 x_3 \vee \bar{x}_3 x_4 \vee x_2 \bar{x}_3 \vee x_2 \bar{x}_4 \vee x_1 x_2 \vee x_1 \bar{x}_3 .$$

После выполнения всех операций поглощения элементарных конъюнкций ранга 3 получаем следующую ДНФ:

$$f^2 = \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_3 x_4 \vee x_2 \bar{x}_3 \vee x_2 \bar{x}_4 \vee x_1 x_2 \vee x_1 \bar{x}_3 .$$

Дальнейшее применение операций неполного склеивания ничего нового не дает. Следовательно, ДНФ f^2 является сокращенной ДНФ функции f.

С целью упрощения вычислений Мак-Класки было предложено более удобное оформление алгоритма Квайна.

7.3.5 Метод Квайна – Мак-Класки

Выразим лежащие в основе метода Квайна операции склеивания и поглощения как операции над троичными векторами, представляющими интервалы булева пространства аргументов $x_1, x_2, \ldots, x_n$ и соответствующие им элементарные конъюнкции.

Соседние векторы $\boldsymbol{u}$ и $\boldsymbol{v}$ (отличающиеся ровно одной компонентой) можно «склеить», образовав новый вектор путем передачи его компонентам совпадающих значений одноименных компонент векторов $\boldsymbol{u}$ и $\boldsymbol{v}$ и присвоения значения «–» той компоненте, по которой эти векторы ортогональны. Например, продукт склеивания соседних векторов $\boldsymbol{u} = 1\,0 - 1\,1 -$ и $\boldsymbol{v} = 1\,0 - 1\,0 -$ представляется вектором $1\,0 - 1 - -$.

Будем далее задавать совершенную ДНФ минимизируемой булевой функции f троичной матрицей $\boldsymbol{U}$. Если в этой матрице существует некоторая пара соседних строк, то их можно «склеить», образовав новую строку, соответствующую продукту склеивания. Добавление в матрицу $\boldsymbol{U}$ получаемой таким образом новой строки называется операцией склеивания. Операция поглощения определяется как удаление из матрицы $\boldsymbol{U}$ некоторой строки, поглощаемой какой-либо из других строк этой матрицы.

При конструировании множества всех простых импликант в методе Квайна много времени тратится на поиск соседних элементарных конъюнкций в преобразуемой ДНФ. Чтобы сократить производимый при этом перебор анализируемых пар конъюнкций, Мак-Класки предложил группировать представляющие их строки не только по рангу, как это делается в методе Квайна, но также по числу единиц, с тем чтобы ограничиться затем рассмотрением таких пар строк, одна из которых содержит на одну единицу больше, чем другая. Нетрудно убедиться в том, что только такие строки могут быть соседними.

Соответственно все множество строк исходной булевой матрицы $\boldsymbol{U}^0$, задающей совершенную ДНФ, разбивается на классы C_i^0 строк, имеющих одно и то же число единиц. В класс C_i^0 попадают строки, имеющие i компонент со значением 1. Строки, входящие в один класс, не могут склеиваться между собой, склеиваться могут только строки классов, индексы i которых отличаются ровно на 1. Продукты склеивания строк булевой матрицы $\boldsymbol{U}^0$ образуют троичную матрицу $\boldsymbol{U}^1$, строки которой также поделены на классы, при этом в класс C_i^1 попадут продукты склеивания строк из классов C_i^0 и C_{i+1}^0. Для классов C_i^1 справедливо все то, что было сказано относительно классов C_i^0. В общем случае строки троичной матрицы $\boldsymbol{U}^j$ представляют элементарные конъюнкции ранга $n-j$. Они разбиваются на классы C_i^j, куда входят строки, содержащие i компонент со значением 1 и j компонент, имеющих значение «–».

После выполнения всех операций неполного склеивания поглощаться могут только те строки, которые участвовали в операциях склеивания. В методе Квайна – Мак-Класки принято помечать такие строки звездочкой, с тем чтобы они могли быть автоматически удалены после выполнения очередной итерации склеиваний. После получения на некотором шаге пустой матрицы $\boldsymbol{U}^p$ непомеченные строки всех классов C_i^j представляют собой простые импликанты искомой сокращенной ДНФ.

На рисунке 7.10 продемонстрирована последовательность вычислений алгоритма Квайна – Мак-Класки на примере уже рассмотренной выше (7.1) булевой функции f четырех аргументов. Ее совершенная ДНФ, принимаемая в качестве отправной, задается на этом рисунке булевой матрицей, разбитой на классы C_1^0, C_2^0, C_3^0, C_4^0. Строки всех классов C_k^j пронумерованы и помечены символом «*», если они участвовали в операциях склеивания с другими строками из C_{k-1}^j или C_{k+1}^j. Для строк C_k^j, являющихся продуктами склеивания, справа указаны номера склеиваемых строк из C_k^{j-1} и C_{k+1}^{j-1}.

Строки, не отмеченные на рисунке 7.10 символом «*», не склеиваются ни с одной другой строкой и, следовательно, задают простые импликанты. Присвоив им индивидуальные имена, получаем сокращенную ДНФ, заданную следующей троичной матрицей:

x_1	x_2	x_3	x_4	
0	–	1	0	*a*
–	–	0	1	*b*
–	1	0	–	*c*
–	1	–	0	*d*
1	–	0	–	*e*
1	1	–	–	*f*

C_1^0 ------------

1	0 0 0 1	*
2	0 0 1 0	*
3	0 1 0 0	*
4	1 0 0 0	*

C_2^0 ------------

5	0 1 0 1	*
6	0 1 1 0	*
7	1 0 0 1	*
8	1 1 0 0	*

C_3^0 ------------

9	1 1 0 1	*
10	1 1 1 0	*

C_4^0 ------------

11	1 1 1 1	*

C_1^1 ------------

12	0 – 0 1	1,5*
13	– 0 0 1	1,7*
14	0 – 1 0	2,6
15	0 1 0 –	3,5*
16	0 1 – 0	3,6*
17	– 1 0 0	3,8*
18	1 0 0 –	4,7*
19	1 – 0 0	4,8*

C_2^1 ------------

20	– 1 0 1	5,9*
21	– 1 1 0	6,10*
22	1 – 0 1	7,9*
23	1 1 0 –	8,9*
24	1 1 – 0	8,10*

C_3^1 ------------

25	1 1 – 1	9,11*
26	1 1 1 –	10,11*

C_1^2 ------------

27	– – 0 1	12,22
28	– 1 0 –	15,23
29	– 1 – 0	16,24
30	1 – 0 –	18,23

C_2^2 -------------

31	1 1 – –	23,26

Рисунок 7.10 – Последовательность вычислений алгоритма Квайна – Мак-Класки

7.3.6 Построение и покрытие матрицы Квайна

Следующим шагом после получения всех простых импликант является нахождение их совокупностей, составляющих безызбыточные множества импликант, имплицирующие исходную функцию. Из безызбыточных ДНФ выбираются кратчайшие и минимальные ДНФ. При отыскании этих форм удобно пользоваться *импликантной таблицей* или *матрицей* (таблицей) *Квайна*, столбцам которой соответствуют элементы множества P, строкам – элементы множества K. Множество K состоит из полных элементарных конъюнкций совершенной ДНФ функции (задающей характеристическое множество функции), а множество P – из простых импликант этой функции. Элемент матрицы на пересечении i-й строки и j-го столбца имеет значение 1, если полная элементарная конъюнкция k_i совершенной ДНФ, приписанная i-й строке, имплицирует простую импликанту p_j, приписанную j-му столбцу.

Таким образом, булева матрица Квайна $\boldsymbol{B}$ задает бинарное отношение формальной импликации между членами совершенной ДНФ и членами сокращенной ДНФ булевой функции. Элемент $b_i^j = 1$, если $k_i \Rightarrow p_j$, и $b_i^j = 0$ в противном случае.

Для рассматриваемой функции матрица Квайна имеет следующий вид:

$$\boldsymbol{B} = \begin{array}{cccccc|c} a & b & c & d & e & f & \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 1 & 1 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 & 1 & 0 & 4 \\ 0 & 1 & 1 & 0 & 0 & 0 & 5 \\ 1 & 0 & 0 & 1 & 0 & 0 & 6 \\ 0 & 1 & 0 & 0 & 1 & 0 & 7 \\ 0 & 0 & 1 & 1 & 1 & 1 & 8 \\ 0 & 1 & 1 & 0 & 1 & 1 & 9 \\ 0 & 0 & 0 & 1 & 0 & 1 & 10 \\ 0 & 0 & 0 & 0 & 0 & 1 & 11 \end{array}$$

Формируя из импликант функции f, соответствующих столбцам матрицы $\boldsymbol{B}$, некоторую ее ДНФ, надо побеспокоиться о том, чтобы для каждого из элементов множества M_f^1 в ДНФ нашелся член, принимающий на нем значение 1. Это означает, что из матрицы $\boldsymbol{B}$ надо выбрать такую совокупность столбцов, что для каждой строки матрицы в выбранной совокупности должен быть хотя бы один столбец, содержащий в этой строке единицу. Стремясь получить кратчайшую ДНФ, находим кратчайшее столбцовое покрытие булевой матрицы $\boldsymbol{B}$ (содержащее минимальное число столбцов).

Ясно, что в любое такое покрытие должны входить все столбцы, в которых содержится единица, являющаяся единственной в некоторой строке. Простые импликанты, обозначающие столбцы с указанным свойством, составляют так называемое *ядро* булевой функции. Они входят во всякую безызбыточную систему простых импликант этой функции. В нашем случае ядро составляют простые импликанты b, a, e и f (покрывающие строки с единственной единицей – 1, 2, 4 и 11).

В общем случае задача поиска кратчайшего покрытия булевой матрицы рассматривалась в пункте 3.3.3. Для данной матрицы задача решается легко, поскольку ряд строк в матрице (1, 2, 4 и 11) содержат лишь по одной единице, и, следовательно, выбор столбцов (b, a, e и f), содержащих эти единицы, обязателен. Эти столбцы, кроме строк 1, 2, 4 и 11, покрывают также и строки 5, 6, 7, 8, 9 и 10. Непокрытой оказалась единственная строка – третья, которая покрывается столбцами *c* и *d*. Таким образом, для нашей булевой матрицы существует два кратчайших покрытия: {*a*, *b*, *c*, *e*, *f*} и {*a*, *b*, *d*, *e*, *f*} и соответственно две кратчайшие ДНФ, составленные из отобранных простых импликант:

0	–	1	0		0	–	1	0
–	–	0	1		–	–	0	1
–	1	0	–		–	1	–	0
1	–	0	–		1	–	0	–
1	1	–	–		1	1	–	–

Эти же ДНФ являются и минимальными. В алгебраическом виде они представляются следующими формулами:

$$D_1 = \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_3 x_4 \vee x_2 \bar{x}_3 \vee x_1 \bar{x}_3 \vee x_1 x_2 ;$$

$$D_1 = \bar{x}_1 x_3 \bar{x}_4 \vee \bar{x}_3 x_4 \vee x_2 \bar{x}_4 \vee x_1 \bar{x}_3 \vee x_1 x_2 .$$

7.4 Визуальный метод минимизации булевых функций

Этот метод основан на визуальном распознавании интервалов булева пространства на его двумерной развертке – карте Карно – и позволяет быстро находить оптимальные или близкие к ним интервальные покрытия характеристического множества M_f^1 булевой функции с небольшим числом аргументов – до шести – десяти.

7.4.1 Представление булевой функции на карте Карно

На карте Карно булева функция $f(x_1, x_2, \dots x_n)$ задается указанием в каждой клетке значения, которое она принимает на соответствующем элементе булева пространства.

Например, на рисунке 7.11 показана булева функция $f(x_1, x_2, x_3, x_4, x_5)$, заданная в векторном виде

1 0 0 0 0 1 0 1 0 0 0 1 1 1 0 1 1 0 0 0 0 1 1 1 1 0 0 0 1 1 1 1.

Характеристическое множество этой функции состоит из следующих наборов значений переменных:

M_f^1 = {0 0 0 0 0, 0 0 1 0 1, 0 0 1 1 1, 0 1 0 1 1, 0 1 1 0 0, 0 1 1 0 1, 0 1 1 1 1, 1 0 0 0 0, 1 0 1 0 1, 1 0 1 1 0, 1 0 1 1 1, 1 1 0 0 0, 1 1 1 0 0, 1 1 1 0 1, 1 1 1 1 0, 1 1 1 1 1}.

Простые импликанты булевой функции *f* представляются максимальными интервалами булева пространства *M*, не пересекающимися с множеством M_f^0.

На карте Карно интервалы булева пространства легко выявляются. При их распознавании полезно помнить, что

– любой элемент n-мерной карты Карно (соответствующей одной клетке) представляет собой интервал n-го ранга (представляющий его троичный вектор содержит n компонент, имеющих значение 1 или 0);

– два интервала i-го ранга, изображения которых на развертке симметричны относительно некоторой оси и полностью находятся в зоне ее симметрии, образуют интервал $(i-1)$-го ранга.

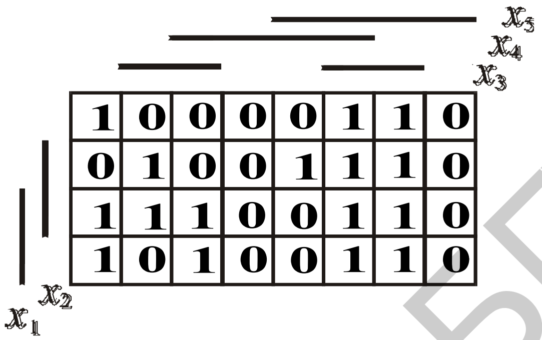


Рисунок 7.11 – Карта Карно полностью определенной булевой функции $f(x_1, x_2, x_3, x_4, x_5)$

Интервал n-го ранга задает полную элементарную конъюнкцию, а интервал i-го ранга совпадает с характеристическим множеством элементарной конъюнкции i-го ранга (содержащей i литералов).

Булева функция, представленная на рисунке 7.11, имеет восемь простых импликант, соответствующие им максимальные интервалы выделены на рисунке 7.12.

7.4.2 Минимизация ДНФ с помощью карт Карно

На карте Карно задача минимизации булевой функции в классе ДНФ сводится к поиску минимального числа максимальных интервалов, покрывающих все элементы булева пространства, составляющие характеристическое множество M_f^1 булевой функции.

Рассмотрим, например, булеву функцию четырех переменных, представленную следующей ДНФ:

$$f(a,b,c,d) = a\,\bar{b}\,\bar{c} \vee a\,c\,\bar{d} \vee \bar{a}\,b\,c \vee \bar{b}\,c\,d\,.$$

Отобразив члены ДНФ соответствующими интервалами пространства переменных a, b, c, d и объединив их, получим изображение функции на карте Карно (рисунок 7.13).

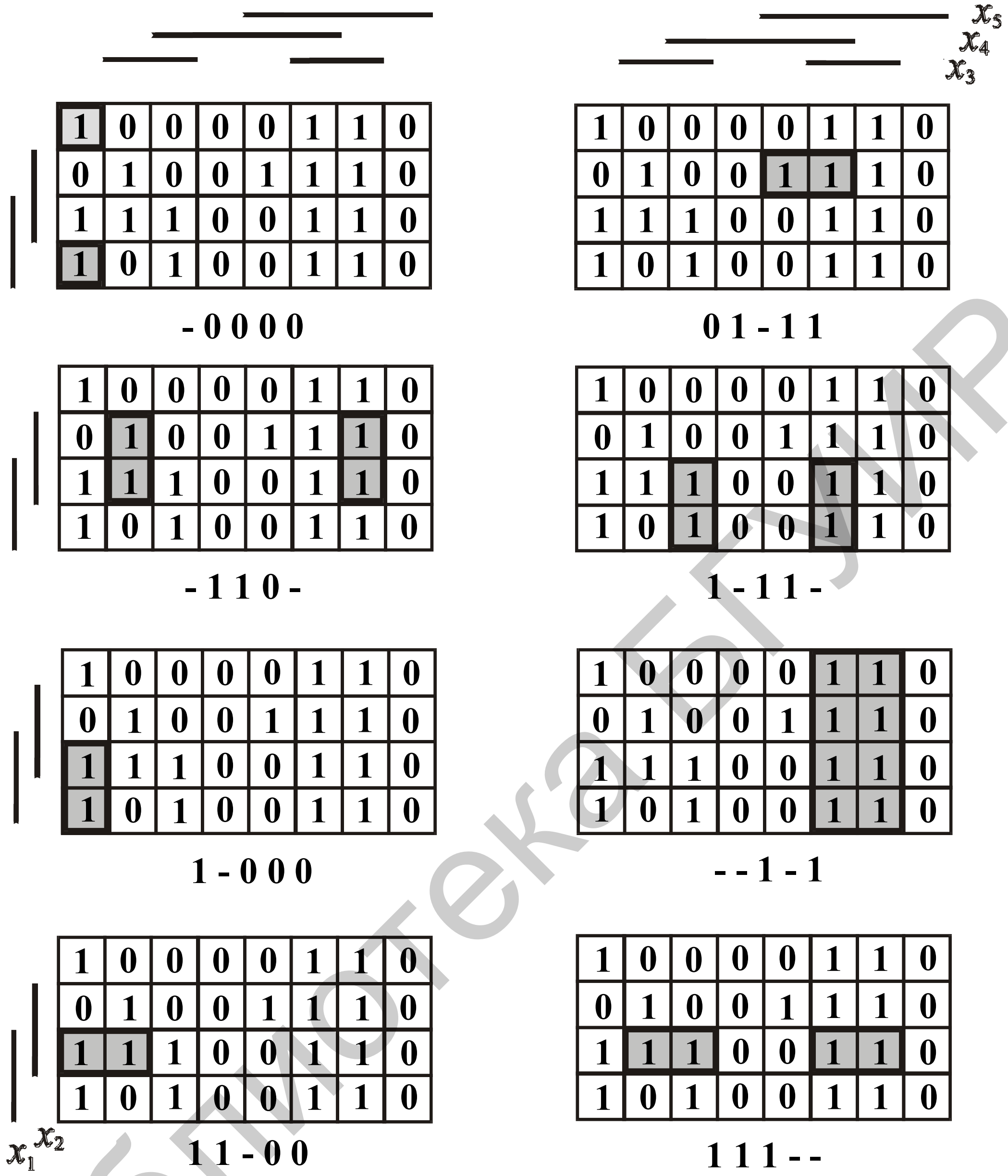


Рисунок 7.12 – Все максимальные интервалы функции

Легко увидеть, что это же изображение можно покрыть тремя интервалами, как показано на рисунке 7.14. Выписав затем соответствующие полученным интервалам элементарные конъюнкции и взяв их дизъюнкцию, найдем кратчайшую ДНФ заданной функции:

$$f(a, b, c, d) = a\,\bar{b} \vee b\,c\,\bar{d} \vee \bar{a}\,c\,d\,.$$

В выборе минимального множества максимальных интервалов, покрывающих все единичные значения булевой функций, и заключается суть визуального метода, использующего человеческие свойства визуального восприятия и распознавания двумерных изображений.

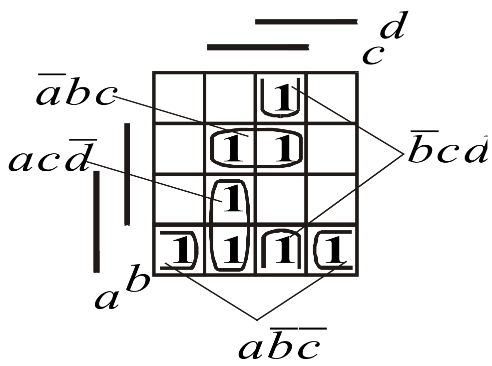


Рисунок 7.13 – Карта Карно булевой функции
$f = a\,\bar{b}\,\bar{c} \vee ac\,\bar{d} \vee \bar{a}\,bc \vee \bar{b}\,cd$

При минимизации более сложных булевых функций (с большим числом переменных) эффективные операции визуального распознавания целесообразно дополнить систематическими процедурами последовательного выбора анализируемых элементов изображения.

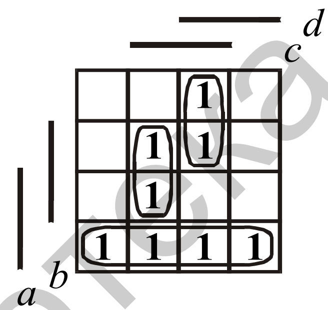


Рисунок 7.14 – Покрытие изображения булевой функции
максимальными интервалами

Для того чтобы найти кратчайшую ДНФ визуальным методом, можно было бы действовать аналогично методам, рассмотренным в подразделе 7.3: найти все максимальные интервалы, затем построить все безызбыточные покрытия ими изображения множества M_f^1 и выбрать среди них кратчайшее. Ниже описываются методы, позволяющие находить точное решение этой задачи, в то же время избегая получения всех максимальных интервалов и тем более построения всех безызбыточных покрытий. Правда, на практике могут встретиться ситуации, когда и эти методы окажутся слишком трудоемкими. В таких случаях можно ограничиться некоторым безызбыточным покрытием, не обязательно кратчайшим.

Число рассматриваемых максимальных интервалов можно сократить на основе следующих рассуждений. Если некоторый элемент ***a*** (булев вектор) множества M_f^1 принадлежит лишь одному из максимальных интервалов этого множества, а именно некоторому интервалу ***u*** (троичный вектор), то очевидно, что любое кратчайшее интервальное покрытие множества M_f^1 будет содержать этот интервал ***u***. В этом случае элемент ***a*** называется *определяющим,* а однозначно определяемый им интервал ***u*** – *обязательным*.

Для того чтобы установить, является ли некоторый элемент ***a*** определяющим, достаточно найти все соседние с ним элементы в множестве M_f^1, а затем построить содержащий их минимальный интервал пространства M_f^1, называемый *минимальным поглощающим интервалом.* Если все элементы этого интервала принадлежат множеству M_f^1, то он является интервалом множества M_f^1, максимальным и притом обязательным, а элемент ***a*** является определяющим. В противном случае ни интервал, ни элемент данными свойствами не обладают.

Векторное представление минимального поглощающего интервала для заданной совокупности элементов булева пространства получается весьма просто. Строится троичный вектор, компоненты которого принимают значения одноименных компонент соответствующих булевых векторов, если эти значения совпадают, и принимают значение «–», если среди этих значений встречается как 0, так и 1.

Например, рассмотрим элемент 0 0 1 1 1 карты Карно (см. рисунки 7.11, 7.12). Он имеет в пространстве M_f^1 три соседних с ним элемента: 0 0 1 0 1, 0 1 1 1 1 и 1 0 1 1 1. Минимальный поглощающий их интервал – – 1 – 1 (состоящий из восьми элементов булева пространства E^5) целиком содержится в множестве M_f^1 (см. карту Карно на рисунке 7.12). Отсюда элемент 0 0 1 1 1 является определяющим, а интервал – – 1 – 1 – обязательным. На карте Карно видно, что элемент 0 0 1 1 1 принадлежит единственному максимальному интервалу и этим интервалом является – – 1 – 1. Все три элемента из M_f^1, соседние с определяющим элементом 0 0 1 1 1, также содержатся в этом интервале.

Элемент 1 1 1 1 0 имеет в пространстве M_f^1 три соседних с ним элемента: 1 1 1 0 0, 1 0 1 1 0 и 1 1 1 1 1. Минимальный поглощающий их интервал 1 – 1 – – (состоящий из восьми элементов булева пространства E^5) содержит элемент 1 0 1 0 0, не входящий в множество M_f^1, следовательно, элемент 1 1 1 1 0 не является определяющим. На карте Карно видно, что элемент 1 1 1 1 0 принадлежит двум максимальным интервалам: 1 1 1 – – (куда он входит со своими соседями 1 1 1 0 0 и 1 1 1 1 1) и 1 – 1 1 – (куда он входит с соседями 1 0 1 1 0 и 1 1 1 1 1).

7.4.3 Многошаговый процесс минимизации с помощью карты Карно

Найдя среди элементов множества M_f^1 хотя бы один обязательный интервал, можно открыть дорогу цепному процессу конструирования элементов покрытия этого множества, который при благоприятных обстоятельствах может завершиться получением точного решения интересующей нас задачи.

Рассмотрим текущую ситуацию, когда часть элементов множества M_f^1 оказывается уже покрытой некоторыми интервалами, выбор которых был строго обоснован, а другая часть, образующая текущее, т. е. изменяемое в процессе решения, множество M^*, состоит из тех элементов множества M_f^1, которые не принадлежат ни одному из выбранных интервалов. Именно об их покрытии и следует далее побеспокоиться.

Пусть некоторый элемент $\boldsymbol{a}$ текущего множества M^* принадлежит сразу нескольким максимальным интервалам множества M_f^1. Рассмотрим пересечения этих интервалов с множеством M^*. Если среди этих пересечений найдется такое, которое содержит остальные в качестве своих подмножеств, то можно считать обоснованным выбор любого соответствующего ему максимального интервала (их может оказаться несколько) как очередного элемента конструируемого кратчайшего покрытия. Очевидно, что этот интервал обеспечивает покрытие всех элементов из M^*, которые только можно покрыть интервалами множества M_f^1, покрывающими одновременно элемент $\boldsymbol{a}$. Назовем такой интервал *допустимым*, обратив внимание на относительный характер этого определения: интервал может оказаться допустимым лишь при данной последовательности предшествующих выборов элементов покрытия. Упомянутый элемент $\boldsymbol{a}$ будет называться в этом случае, как и прежде, определяющим.

Можно предложить несколько способов нахождения определяющих элементов в промежуточных ситуациях, когда часть множества M_f^1 уже покрыта. Рассмотрим один из них, основанный на последовательном переборе элементов множества M_f^1 и их анализе. Будем говорить, что некоторые два элемента множества M_f^1 *совместимы,* если поглощающий их минимальный интервал содержится в множестве M_f^1. Элемент $\boldsymbol{a}$ оказывается определяющим, если минимальный поглощающий интервал для всех элементов множества M^*, совместимых с элементом $\boldsymbol{a}$, содержится в множестве M_f^1. Описываемый метод реализует проверку этого отношения.

Перебор в множестве M^*, производимый при поиске совместимых с $\boldsymbol{a}$ элементов, ускоряется тем, что он ограничивается рассмотрением элементов, принадлежащих минимальному поглощающему интервалу $I(\boldsymbol{a})$ для всех соседей элемента $\boldsymbol{a}$ в множестве M_f^1. Очевидно, что совместимые с $\boldsymbol{a}$ элементы могут находиться только в этом интервале. Если минимальный поглощающий интервал $I^*(\boldsymbol{a})$ для всех элементов $\boldsymbol{b} \in I(\boldsymbol{a})$ и $\boldsymbol{b} \in M^*$, совместимых с элементом $\boldsymbol{a}$, содержится в множестве M_f^1, то элемент $\boldsymbol{a}$ является определяющим. Допустимым в этом случае оказывается любой максимальный интервал множества M_f^1, содержащий найденный минимальный поглощающий интервал $I^*(\boldsymbol{a})$ для совместимых с $\boldsymbol{a}$ элементов множества M^*. В частности, эти два интервала могут совпасть.

Обратим внимание на существенное отличие промежуточных ситуаций от начальной, заключающееся в том, что в начальной ситуации от определяющего элемента $\boldsymbol{a}$ требуется, чтобы в множестве M_f^1 содержался минимальный поглощающий интервал для всех совместимых с $\boldsymbol{a}$ элементов множества M_f^1, в то время как в текущей ситуации это требование ослаблено: там рассматривается минимальный поглощающий интервал только для тех совместимых с $\boldsymbol{a}$ элементов, которые принадлежат текущему множеству M^*.

Это отличие и определяет цепной характер процесса конструирования кратчайшего интервального покрытия множества M_f^1. Включение в покрытие очередного допустимого интервала сопровождается соответствующим сокращением множества M^*, а это, в свою очередь, может привести к тому, что некоторые из остающихся в множестве M^* элементов могут стать определяющими. Последнее позволит включить в решение новые допустимые интервалы, далее сократить текущее множество M^* и т. д.

На рисунке 7.15 приведен процесс конструирования кратчайшего интервального покрытия изображения множества M_f^1 для приведенной выше функции (см. рисунок 7.11). Определяющие элементы выделены курсивом.

Процесс поиска определяющих элементов начнем с первого элемента 0 0 0 0 0 матричного представления булевой функции $f(x_1, x_2, x_3, x_4, x_5)$. Он имеет одного соседа – 1 0 0 0 0. Минимальный поглощающий интервал для элементов 0 0 0 0 0 и 1 0 0 0 0 в множестве M_f^1 $I(00000) = –0000$. Так как этот интервал содержится в пространстве M_f^1, то он является обязательным, а элемент 0 0 0 0 0 – определяющим. А раз так, то интервал – 0 0 0 0 принимается в качестве первого элемента решения. Элементы, принадлежащие найденному интервалу, исключаются из дальнейшего рассмотрения. Текущее покрываемое множество M^* сокращается до

$$M^* = \{00101, 00111, 01011, 01100, 01101, 01111, 10101, 10110,$$
$$10111, 11000, 11100, 11101, 11110, 11111\}.$$

Следующий непокрытый элемент 0 0 1 0 1 имеет трех соседей (0 0 1 1 1, 0 1 1 0 1 и 1 0 1 0 1). Минимальный поглощающий интервал для элемента 0 0 1 0 1 и его соседей в множестве M_f^1 $I(0\,0\,1\,0\,1) = --1-1$. Все элементы этого интервала содержатся в множестве M*, поэтому минимальный поглощающий интервал $--1-1$ является допустимым, а сам элемент $0\,0\,1\,0\,1$ определяющим. Интервал $--1-1$ включается в решение в качестве второго элемента, а принадлежащие ему восемь элементов исключаются из множества M* подлежащих покрытию элементов:

$$M^* = \{0\,1\,0\,1\,1,\ 0\,1\,1\,0\,0,\ 1\,0\,1\,1\,0,\ 1\,1\,0\,0\,0,\ 1\,1\,1\,0\,0,\ 1\,1\,1\,1\,0\}.$$

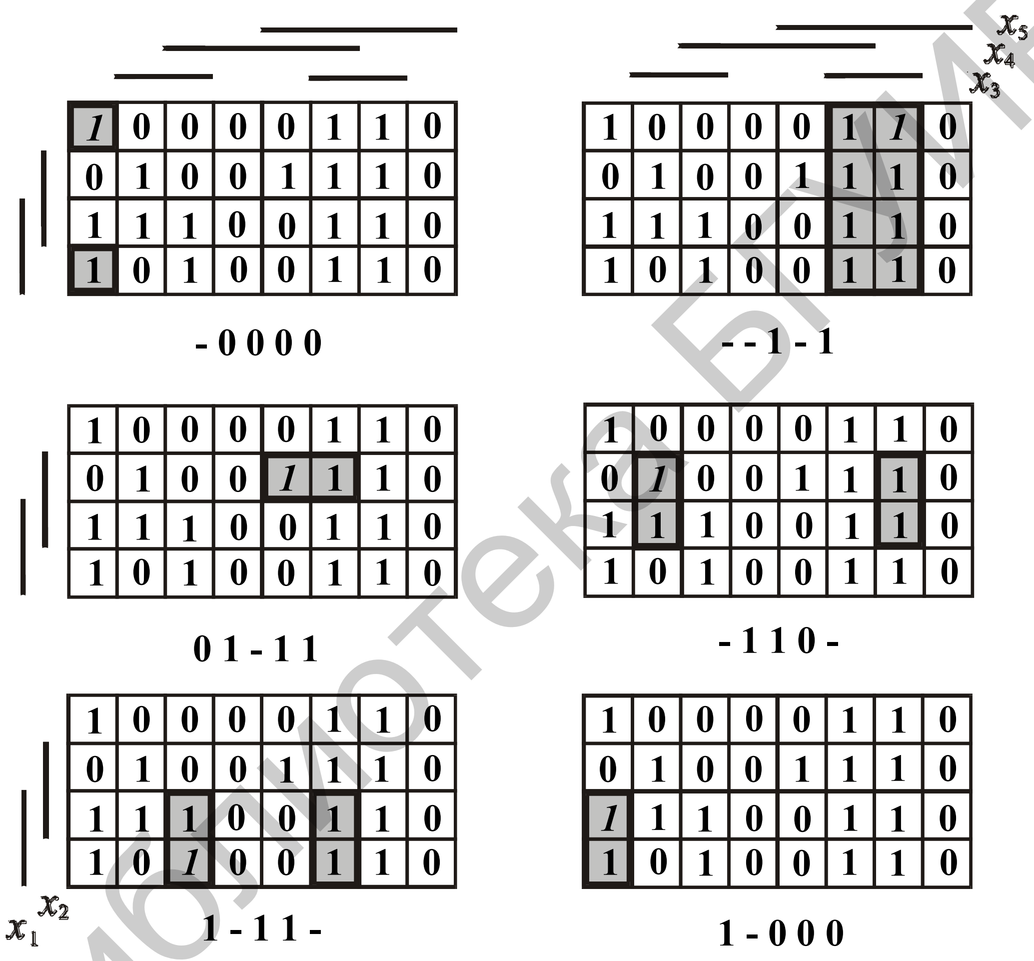


Рисунок 7.15 – Разложение на интервалы изображения функции

Минимальным поглощающим интервалом для элемента $0\,1\,0\,1\,1$, имеющего одного соседа $0\,1\,1\,1\,1$ в множестве M_f^1, является $I(0\,1\,0\,1\,1) = 0\,1-1\,1$. В этом интервале нет ни одного совместимого с $0\,1\,0\,1\,1$ элемента из M*. Но так как интервал $I(0\,1\,0\,1\,1) = 0\,1-1\,1$ содержится в пространстве M_f^1, то его можно выбрать в качестве допустимого для $0\,1\,0\,1\,1$.

Теперь очередь доходит до элемента $0\,1\,1\,0\,0$ и соседних с ним в множестве M_f^1 элементов $1\,1\,1\,0\,0$ и $0\,1\,1\,0\,1$. Находим, что поглощающий их интервал

$I(01100) = -110-$ содержит один совместимый с 01100 элемент из M^* – 11100 (совместимый, так как $-1100 \in M_f^1$). В качестве допустимого можно взять этот интервал -1100, но он не максимальный в M_f^1, максимальным является интервал $-110-$. Соответственно 01100 является определяющим, а интервал $-110-$ допустимым. Множество M^* сокращается:

$$M^* = \{10110, 11000, 11110\}.$$

Элемент 10110 обладает соседями 11110 и 10111 в множестве M_f^1, минимальным поглощающим их интервалом является $I(10110) = 1-11-$. В этом интервале содержится один совместимый с 10110 элемент из M^* – 11110. В качестве допустимого выбирается интервал $1-11-$, максимальный в M_f^1. В результате в множестве M^* остается единственный непокрытый элемент – 11000. Он обладает двумя соседями в множестве M_f^1 – 11100 и 10000, минимальный поглощающий их интервал – $I(11000) = 1--00$. Этот интервал не содержится целиком в M_f^1 (на одном из его элементов 10100 функция принимает значение 0). Следовательно, интервал $1--00$ не является допустимым, а в качестве допустимого можно взять один из максимальных в M_f^1 интервалов ($1-000$ или $11-00$), содержащих 11000, например $1-000$. В результате все элементы множества M_f^1 окажутся покрытыми ($M^* = \varnothing$).

Таким образом, повторяя описанные шаги при минимизации булевой функции, изображенной на рисунке 7.11, получаем кратчайшее интервальное покрытие множества M_f^1, что приводит к кратчайшей ДНФ булевой функции:

x_1	x_2	x_3	x_4	x_5
–	0	0	0	0
–	–	1	–	1
0	1	–	1	1
–	1	1	0	–
1	–	1	1	–
1	–	0	0	0

В рассмотренном примере процесс последовательного нахождения в характеристическом множестве M_f^1 определяющих элементов и определяемых ими допустимых интервалов благополучно завершился получением кратчайшего интервального покрытия этого множества. Однако в других примерах могут встретиться критические ситуации, когда в текущем множестве M^* не удастся найти ни одного определяющего элемента.

В таком случае для получения точного решения потребуется выполнить некоторый перебор различных вариантов построения покрытия, соответствующих выбору различных максимальных интервалов в качестве очередного элемента покрытия. Рассмотрение различных вариантов выбора очередного элемента покрытия из множества возможных образует точку ветвления процесса поиска кратчайшего интервального покрытия. Число исходящих из этой точки ветвей равно числу вариантов выбора.

Например, рассмотрим для рассмотренной выше функции (см. рисунок 7.11) элемент 1 1 1 0 0. Он принадлежит трем максимальным интервалам, показанным на рисунке 7.16:

$$1\,1\,1 - -,\quad -\,1\,1\,0\,-,\quad 1\,1 - 0\,0.$$

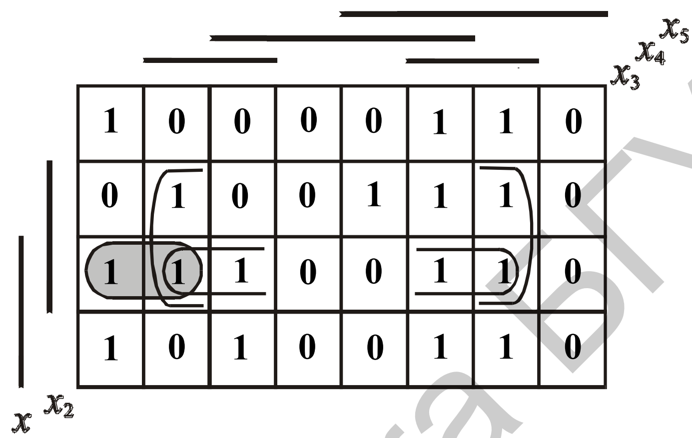


Рисунок 7.16 – Максимальные интервалы, которым принадлежит элемент 1 1 1 0 0

Очевидно, что чем меньше соседей в множестве M_f^1 имеет некоторый элемент этого множества, тем меньшему, как правило, числу различных максимальных интервалов он принадлежит. Учитывая это обстоятельство, предлагается начинать построение интервального покрытия множества M_f^1 с поиска интервалов для покрытия элементов с минимальным числом соседей – в этом случае конструирование покрытия будет проходить быстрее.

Например, построим кратчайшее интервальное покрытие множества M_f^1 приведенной выше функции (см. рисунок 7.11), следуя методу минимального соседства. Элементов, не имеющих соседей, в множестве M_f^1 не оказывается, поэтому начнем с рассмотрения элементов, имеющих по одному соседу. Таких элементов два: 0 0 0 0 0 и 0 1 0 1 1. Для них находятся максимальные интервалы – 0 0 0 0 и 0 1 – 1 1, принадлежащие им четыре элемента исключаются из дальнейшего рассмотрения. Далее выбираем среди оставшихся элементов множества M^* элементы с двумя соседями (0 1 1 0 0, 1 0 1 1 0, 1 1 0 0 0), тремя (0 0 1 0 1) и т. д. Выбирая в множестве M^* элементы с минимальным числом соседей и включая в решение покрывающие их максимальные интервалы, осуществим показанное на первых шести картах Карно (см. рисунок 7.12) разложение множества M_f^1 на интервалы, что приведет нас к уже знакомому решению.

Задания

1. Проверить основные элементарные функции алгебры логики на принадлежность замкнутым классам.

2. Определить, каким замкнутым классам принадлежат функции:

$f_1 = xy \oplus z$;

$f_2 = (x \oplus y) \sim z$;

$f_3 = (x \to y) \& z$;

$f_4 = (x \sim z) \oplus y$;

$f_5 = (x \to y) \& z$;

$f_6 = (x \oplus z) \to y$.

Проверить на полноту системы функций:

$\{\bar{x}\ \ \bar{y}\}$;

$\{\bar{x}, x \vee y\}$;

$\{\bar{x}, x \wedge y\}$;

$\{1, x\,y, x \oplus y\}$;

$\{x \to y,\ \bar{x}\}$;

$\{x \to y, x \oplus y\}$;

$\{x \vee y,\ x \sim y, 1\}$;

$\{0, x\,y, x \sim y\}$;

$\{x\,y, x \oplus y \oplus 1\}$;

$\{x \to y, 0\}$;

$\{x \downarrow y\}$.

3. Найти минимальные ДНФ функций методом Квайна – Мак-Класки и визуальным методом на карте Карно:

$f_1(x) = 0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,0\,0\,1$;

$f_2(x) = 1\,1\,0\,0\,0\,1\,1\,1\,0\,1\,1\,0\,1\,0\,0\,1$;

$f_3(x) = 1\,0\,1\,1\,0\,0\,1\,0\,0\,0\,1\,1\,1\,0\,0\,0$;

$f_4(x) = 1\,0\,1\,1\,0\,0\,1\,0\,0\,0\,1\,1\,1\,0\,0\,0$;

$f_5(x) = 0\,1\,0\,0\,1\,1\,0\,0\,0\,1\,0\,0\,1\,0\,1\,1$;

$f_6(x) = 0\,1\,0\,1\,0\,0\,0\,1\,0\,1\,0\,1\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1\,0\,0\,0\,1\,1\,1\,0\,0\,0$;

$f_7(x) = 1\,0\,0\,0\,1\,0\,1\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1\,0\,0\,0\,1\,1\,1\,0\,0\,0\,0\,1\,0$;

$f_8(x,y,z,t) = xy\bar{z} \vee x\bar{y}\bar{z} \vee y\bar{z}t \vee \bar{x}\bar{y}zt \vee x\bar{y}\bar{z}\bar{t} \vee x\bar{z}t \vee \bar{x}y\bar{z}\bar{t} \vee \bar{x}\bar{y}\bar{z} \vee xy\bar{z}t$;

$f_9(x,y,z,t) = \bar{x}yz \vee \bar{y}\bar{z} \vee xyzt \vee \bar{x}\bar{y}zt \vee \bar{x}\bar{y}\bar{z}\bar{t} \vee x\bar{z}t$.

4. Построить схемы из вентилей И-НЕ, реализующие ДНФ функций из задания 3.

ЛИТЕРАТУРА

О с н о в н а я

1. Лекции по теории графов / В. А. Емеличев [и др.]. – М. : Наука, 1990. – 384 с.

2. Закревский, А. Д. Логические основы проектирования дискретных устройств / А. Д. Закревский, Ю. В. Поттосин, Л. Д. Черемисинова. – М. : Физматлит, 2007. – 589 с.

3. Закревский, А. Д. Основы логического проектирования. В 2 кн. Кн. 1 : Комбинаторные алгоритмы дискретной математики / А. Д. Закревский, Ю. В. Поттосин, Л. Д. Черемисинова. – Минск : ОИПИ НАН Беларуси, 2004. – 226 с.

4. Закревский, А. Д. Основы логического проектирования. В 2 кн. Кн. 2 : Оптимизация в булевом пространстве / А. Д. Закревский, Ю. В. Поттосин, Л. Д. Черемисинова. – Минск : ОИПИ НАН Беларуси, 2004. – 240 с.

5. Новиков, Ф. А. Дискретная математика для программистов / Ф. А. Новиков. – СПб. : Питер, 2005. – 364 с.

Д о п о л н и т е л ь н а я

6. Андерсон, Дж. А. Дискретная математика и комбинаторика / Дж. А. Андерсон. – М. : Изд. дом «Вильямс», 2003. – 958 с.

7. Ахо, Ф. Структуры данных и алгоритмы / Ф. Ахо, Дж. Хопкрофт, Дж. Ульман. – М. : Изд. дом «Вильямс», 1979. – 536 с.

8. Басакер, Р. Конечные графы и сети / Р. Басакер, Т. Саати. – М. : Наука, 1974. – 368 с.

9. Берж, К. Теория графов и ее применения / К. Берж. – М. : ИЛ, 1962. – 320 с.

10. Бондаренко, М. Ф. Компьютерная дискретная математика / М. Ф. Бондаренко, Н. В. Белоус, А. Г. Руткас. – Харьков : Компания СМИТ, 2004. – 480 с.

11. Верещагин, Н. К. Лекции по математической логике и теории алгоритмов. В 3 ч. Ч. 1 : Начала теории множеств / Н. К. Верещагин, А. Шень. – М. : МЦНМО, 2008. – 128 с.

12. Виленкин, Н. Я. Комбинаторика / Н. Я. Виленкин, А. Н. Виленкин, П. А. Виленкин. – М. : ФИМА, МЦНМО, 2006. – 400 с.

13. Гаврилов, Г. П. Сборник задач по дискретной математике / Г. П. Гаврилов, А. А. Сапоженко. – М. : Наука, 1977. – 368 с.

14. Гиндикин, С. Г. Алгебра логики в задачах / С. Г. Гиндикин. – М. : Наука, 1972. – 288 с.

15. Гладков, Л. А. Дискретная математика / Л. А. Гладков, В. В. Курейчик, В. М. Курейчик. – М. : Физматлит, 2014. – 496 с.

16. Глушков, В. М. Синтез цифровых автоматов / В. М. Глушков. – М. : Физматгиз, 1962. – 476 с.

17. Гэри, М. Вычислительные машины и труднорешаемые задачи / М. Гэри, Д. Джонсон. – М. : Мир, 1982. – 416 с.

18. Евстигнеев, В. А. Применение теории графов в программировании / В. А. Евстигнеев. – М. : Наука, 1985. – 352 с.

19. Евстигнеев, В. А. Задачи и упражнения по теории графов и комбинаторике / В. А. Евстигнеев, Л. С. Мельников. – Новосибирск : Изд. НГУ, 1981. – 88 с.

20. Ерусалимский, Я. Н. Дискретная математика: теория, задачи, приложения / Я. Н. Ерусалимский. – М. : Вузовская книга, 2000. – 280 с.

21. Закревский, А. Д. Логический синтез каскадных схем / А. Д. Закревский. – М. : Наука, 1981. – 414 с.

22. Зыков, А. А. Основы теории графов / А. А. Зыков. – М. : Наука, 1987. – 384 с.

23. Кристофидес, Н. Теория графов. Алгоритмический подход / Н. Кристофидес. – М. : Мир, 1978. – 432 с.

24. Кузнецов, О. П. Дискретная математика для инженеров / О. П. Кузнецов, Г. М. Адельсон-Вельский. – М. : Энергия, 1988. – 480 с.

25. Липский, В. Комбинаторика для программистов / В. Липский. – М. : Мир, 1998. – 214 с.

26. Микони, С. В. Дискретная математика для бакалавра: множества, отношения, функции, графы : учеб. пособие / С. В. Микони. – СПб. : Лань, 2012. – 192 с.

27. Миллер, Р. Теория переключательных схем. В 2 т. Т. 1 / Р. Миллер. – М. : Наука, 1970. – 416 с.

28. Оре, О. Теория графов / О. Оре. – М. : Наука, 1980. – 336 с.

29. Рейнгольд, Э. Комбинаторные алгоритмы: теория и практика / Э. Рейнгольд, Ю. Нивергельт, Н. Део. – М. : Мир, 1980. – 476 с.

30. Свами, М. Графы, сети и алгоритмы / М. Свами, К. Тхуласираман. – М. : Мир, 1984. – 455 с.

31. Тишин, В. В. Дискретная математика в примерах и задачах / В. В. Тишин. – СПб. : БХВ-Петербург, 2016. – 336 с.

32. Уилсон, Р. Введение в теорию графов / Р. Уилсон. – М. : Мир, 1977. – 205 с.

33. Хаггарти, Р. Дискретная математика для программистов / Р. Хаггарти. – Техносфера, 2004. – 320 с.

34. Харари, Ф. Теория графов / Ф. Харари. – М. : Мир, 1973. – 304 с. ; 3-е изд. – М. : КомКнига, 2006. – 296 с.

35. Харари, Ф. Перечисление графов / Ф. Харари, Э. Палмер. – М. : Мир, 1977. – 324 с.

36. Холл, М. Комбинаторика / М. Холл. – М. : Мир, 1970. – 424 с.

37. Яблонский, С. В. Введение в дискретную математику / С. В. Яблонский. – М. : Наука, 1986. – 384 с.

38. Stanat, D. F. Discrete mathematics in computer science / D. F. Stanat, D. F. McAllister. – N.J. : Prentice-Hall, inc., Englewood Cliffs, 1977.

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

К

Л

М

Ц

Ч

Э

Я

Св. план 2018, поз. 10

Учебное издание

Черемисинова Людмила Дмитриевна

ДИСКРЕТНАЯ МАТЕМАТИКА

УЧЕБНОЕ ПОСОБИЕ

Редактор *Е. И. Костина*
Корректор *Е. Н. Батурчик*
Компьютерная правка, оригинал-макет *О. И. Толкач*

Подписано в печать 15.03.2019. Формат 60×84 1/16. Бумага офсетная. Гарнитура «Таймс».
Отпечатано на ризографе. Усл. печ. л. 17,55. Уч.-изд. л. 19,0. Тираж 300 экз. Заказ 376.

Издатель и полиграфическое исполнение: учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники».
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий №1/238 от 24.03.2014,
№2/113 от 07.04.2014, №3/615 от 07.04.2014.
ЛП №02330/264 от 14.04.2014.
220013, Минск, П. Бровки, 6