ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

Белгородский государственный технологический университет
им. В. Г. Шухова

Д. А. Куценко, Д. В. Терехов

Математическая логика и теория алгоритмов

Утверждено ученым советом университета в качестве учебного пособия для студентов специальности 230105 — Программное обеспечение вычислительной техники и автоматизированных систем

Белгород
2009

УДК [510.5+510.6](075.8)
ББК 22.12я73
К88

Р е ц е н з е н т ы:

Кандидат физико-математических наук, доцент кафедры высшей математики БГТУ им. В. Г. Шухова *Е. Н. Сергиенко*

Кандидат технических наук, заместитель директора по научной работе ФГУП ВИОГЕМ *С. С. Серый*

Куценко, Д. А.

К88 Математическая логика и теория алгоритмов: учеб. пособие / Д. А. Куценко, Д. В. Терехов. — Белгород: Изд-во БГТУ, 2009. — 64 с.

В учебном пособии изложены основы математической логики и теории алгоритмов. Приведены основные сведения из логики высказываний, логики предикатов, логических исчислений, формальной арифметики, рассмотрены метод резолюций, машина Тьюринга, язык Brainfuck. Материал соответствует основной части лекционного курса дисциплины «Математическая логика и теория алгоритмов». В конце каждой главы приведены примеры решения типовых задач.

Для студентов, обучающихся по специальности 230105 — Программное обеспечение вычислительной техники и автоматизированных систем.

Издание публикуется в авторской редакции.

УДК [510.5+510.6](075.8)
ББК 22.12я73

Оглавление

Введение

Логикой[1)] называют науку о законах и формах мышления. Наиболее важной её частью является *формальная логика*, изучающая формы правильных рассуждений. *Математическая логика* — это раздел формальной логики, посвящённый формам рассуждений, принятым в математике. Объектом изучения здесь являются математические доказательства и вопросы оснований математики.

Логика, как искусство рассуждений, зародилась в глубокой древности. Первый этап развития формальной логики, длившийся два тысячелетия, называется *традиционной логикой.* Со второй половины XIX в. начала бурно развиваться математическая логика. Именно использование математических методов отличает современную логическую науку от традиционной. Впоследствии математическая логика нашла широкое применение в самых разных областях человеческой деятельности. Это одна из дисциплин, составляющих теоретический фундамент информатики.

Предмет математической логики разнообразен. Современная математическая логика состоит из множества разделов, но во все учебники и учебные пособия всегда входят *логика высказываний, исчисление высказываний* и *логика предикатов.* Практическую значимость имеет также *исчисление предикатов.* В последние несколько десятилетий бурное развитие получила *теория алгоритмов,* имеющая много общего с методологией математической логики. Часто эти две научные дисциплины объединяют в одну, называемую *«Математическая логика и теория алгоритмов».*

Четыре вышеуказанных раздела математической логики и один из разделов теории алгоритмов (машина Тьюринга) и составляют данное учебное пособие. Так как пользоваться им будут будущие инженеры-программисты, особое внимание уделено практическим вопросам — включены такие дополнительные разделы, как релейно-контактные схемы, метод резолюций, язык Brainfuck. В конце каждой главы приведены решения типовых задач.

Предполагается, что студент, начинающий работу с данным учебным пособием, знаком с основными понятиями теории множеств и отношений, а также с аппаратом булевых функций.

[1)] от *греч.* λόγος — речь, рассуждение, мысль.

Глава 1

Логика высказываний

1.1. Высказывания

Исходным понятием математической логики является простое логическое высказывание. *Высказывание* — повествовательное предложение какого-либо языка (естественного или искусственного), рассматриваемое лишь с точки зрения того, истинно оно или ложно. Иначе говоря, каждому высказыванию соответствует известное *истинностное значение.* Если некоторой фразе нельзя сопоставить определённое истинностное значение, то она не является высказыванием. В традиционной двузначной логике рассматривается два истинностных значения: «истина» и «ложь». Истинностные значения будем обозначать так: «1» — «истина», «0» — «ложь»[1)]. Например, фразы «Нью-Йорк — столица США» и «$2 + 2 = 4$» — высказывания. Истинностное значение первого — 0 (ложь), истинностное значение второго — 1 (истина). Фразы «Который час?» и «$\sqrt{25}$» высказываниями не являются.

В ряде случаев истинность высказывания зависит от того, какую конкретную реальность (систему, процесс, явление) мы пытаемся описать. В таком случае говорят, что данное высказывание истинно (или ложно) в данном контексте или интерпретации. Например, высказывание «$1 + 1 = 10$» ложно в десятичной системе счисления, но истинно в двоичной. Далее будем считать, что контекст задан и высказывания имеют определённые истинностные значения.

Высказывание называется *простым* или *элементарным*, если его нельзя разложить на части, которые также являются высказываниями. В противном случае оно называется *сложным* или *составным.* В приведённых выше примерах все высказывания являются простыми. Высказывание «5 — нечётное число, меньшее 10» является сложным, так как его можно разложить на высказывания «5 — нечётное число» и «5 меньше 10».

[1)] Обозначения «1» и «0» взяты из дискретной математики и программирования ЭВМ, где широко используется двоичная система счисления.

При формальном подходе понятие «простое высказывание» замещается понятием *«пропозициональная*[2] *переменная»*. Область значений таких переменных состоит из всевозможных высказываний. Пропозициональные переменные будем обозначать заглавными латинскими буквами A, B, C и т. д., возможно с индексами, например X_2. Пусть A — «Марс — планета», а B — «У прямоугольного треугольника все стороны равны». Тогда $A = 1$, а $B = 0$.

1.2. Пропозициональные связки

Сложные высказывания строятся из простых с помощью *пропозициональных связок* «&», «$\vee$», «$\to$», «$\leftrightarrow$», «$\overline{}$»:

$$A \,\&\, B, \quad A \vee B, \quad A \to B, \quad A \leftrightarrow B, \quad \overline{A}.$$

Связь, выраженная с помощью пропозициональных связок, предполагается не обязательно по смыслу, а берётся только по истинностному значению.

Связка «&» соответствует союзам «и», «а», «но» естественного языка и называется *конъюнкцией*[3]. Сложное высказывание, образованное из простых с помощью конъюнкции, истинно только в том случае, когда все составляющие его высказывания истинны. В других случаях оно ложно.

Связка «$\vee$» соответствует союзам «или», «либо» и называется *дизъюнкцией*[4]. Сложное высказывание, образованное из простых с помощью дизъюнкции, истинно, когда хотя бы одно из составляющих его высказываний истинно. Ложно оно только в том случае, когда все составляющие его высказывания ложны[5].

[2] от *лат.* propositio — предложение, суждение, высказывание.

[3] от *лат.* conjunctio — союз, связь. Знак «&» (амперса́нд, *англ.* ampersand) — графическое сокращение латинского союза et — «и». Это отчётливо видно в другом варианте написания: *&*. Название произошло от фразы and, per se and — «и, сама по себе „и“».

[4] от *лат.* disjunctio — разобщение, разделение, различие. Знак «$\vee$» (клин) происходит от первой буквы латинского союза vel — «или».

[5] Здесь и далее под связкой «$\vee$» подразумевается *соединительно-разъединительная дизъюнкция,* обозначающая союз «или» в неисключающем смысле ($A \vee B$ — «или A, или B, или то и другое вместе», т. е. истинность одного высказывания не исключает истинности другого). *Исключающая (строгая) дизъюнкция* ($A \oplus B$ — «либо A, либо B») используется редко и заменяется на $(A \vee B)\ \&\ \overline{A\ \&\ B}$ или $\overline{A \leftrightarrow B}$.

Связка «$\to$» выражает причинно-следственную связь между двумя высказываниями и называется *импликацией*[6]. Импликация играет особую роль в математической логике. Сложное высказывание, образованное из двух простых с помощью импликации, называется *условным.* В условном высказывании $A \to B$ высказывание A называется *антецедентом*[7], а высказывание B — *консеквентом*[8]. Условное высказывание ложно только в том случае, когда его антецедент истинен, а консеквент ложен («из истины не может следовать ложь»). В остальных случаях оно истинно («из истины может следовать только истина», «из лжи может следовать всё что угодно»). Запись $A \to B$ соответствует высказываниям «если A, то B», «A, следовательно B», «A влечёт B», «B, если A», «при наличии A следует B», «B при условии, что A».

Важно проводить различие между достаточными и необходимыми условиями наступления какого-то факта, события, отражаемого в условном высказывании. Условия являются *достаточными,* если при их выполнении всегда наступает данное событие. Условия являются *необходимыми,* если без их выполнения данное событие никогда не наступает. Из определения истинностного значения импликации можно установить, что антецедент является достаточным условием для консеквента, а консеквент — необходимым условием для антецедента. Поэтому запись $A \to B$ также соответствует высказываниям «для A необходимо B», «для B достаточно A», «необходимым условием A является B», «достаточным условием B является A».

Союз «и» также будет соответствовать импликации, а не конъюнкции, если им выражается причинно-следственная связь. Например, сравните: «Он испугался и выстрелил» и «Он выстрелил и испугался»[9].

Связка «$\leftrightarrow$» называется *эквивалентностью*[10]. Запись $A \leftrightarrow B$ соответствует высказываниям «A тогда и только тогда, когда B», «A равносильно B», «Для A необходимо и достаточно B», «A, если и только если B». Сложное высказывание, образованное с помощью эквивалентности, истинно только в том случае, когда истинностные

[6] от *лат.* implicite — тесно связываю.

[7] от *лат.* antecedens — предшествующий, предыдущий.

[8] от *лат.* consequens — следствие, последующий вывод.

[9] В этом случае нарушается закон коммутативности (см. стр. 11).

[10] от *лат.* æqualis — равный и valentis — имеющий силу; равносильность.

значения входящих в него простых высказываний равны. В противном случае оно ложно.

Наконец, «$\overline{A}$» понимается как «не A», «не верно, что A», «A не имеет места» и называется *отрицанием* высказывания A. В классической математической логике считается, что высказывание $\overline{A}$ ложно, когда высказывание A истинно, и истинно, когда высказывание A ложно.

1.3. Основные законы логики

Логику называют *классической,* если она основывается на следующих четырёх законах:

1. *Закон тождества*[11]: каждое высказывание при повторении должно иметь одно и тоже определённое устойчивое содержание.

2. *Закон (запрета) противоречия*[12]: любое высказывание и его отрицание об одном и том же вместе не могут быть истинными.

3. *Закон исключённого третьего*[13]: либо данное высказывание ложно, либо его отрицание ложно, третьего не дано.

4. *Закон достаточного основания*[14]: всякое истинное высказывание должно быть обосновано другими высказываниями, истинность которых доказана.

Первые три закона были сформулированы ещё Аристотелем[15], четвёртый выдвинут Лейбницем[16].

Если какой-либо из перечисленных законов не выполняется, то логику называют *неклассической.* Одним из примеров неклассической логики является трёхзначная логика, предложенной в 1920 г.

---

[11] *лат.* lex identitatius.

[12] *лат.* lex contradictionis.

[13] *лат.* lex exclusi tertii sive medii inter duo contradictoria.

[14] *лат.* lex rationis determinantis sive sufficientis.

[15] Аристо́тель (Ἀριστοτέλης; 384 до н. э.—322 до н. э.) — древнегреческий философ и учёный, основоположник формальной логики.

[16] Готфрид Вильгельм фон Ле́йбниц (Gottfried Wilhelm von Leibniz; 1646—1716) — немецкий (саксонский) философ, математик, юрист, дипломат.

Я. Лукасевичем[17]. В этой логике не выполняется закон исключённого третьего, так как помимо истинностных значений «истина» и «ложь» в ней имеется третье значение — «возможно». Неклассические логики в данном пособии рассматриваться не будут.

1.4. Алгебра логики

Поскольку традиционная логика под влиянием математики превратилась в математическую логику, то естественным было использование в логике терминологии, принятой в математике.

В частности, вводятся понятия «пропозициональная переменная», «логическая операция», «алгебра логики», «формула алгебры логики». Выше было дано определение понятия «пропозициональная переменная». Понятию «пропозициональная связка» соответствует понятие *«логическая операция»*. Множество пропозициональных переменных $\mathscr{P} = \{A, B, C, \ldots\}$ с заданными над ним логическими операциями $\mathscr{F} = \{\overline{\ }, \&, \vee, \to, \leftrightarrow\}$ формируют *алгебру*[18] *логики,* т. е. упорядоченную пару

$$\mathscr{A}_{\text{Л}} = \langle \mathscr{P}, \mathscr{F} \rangle.$$

Алгебру логики называют также *алгеброй высказываний.*

Всякому сложному высказыванию, которое может быть получено из элементарных высказываний посредством применения логических операций, соответствует *формула*[19] алгебры логики. Формулы будем обозначать готическими буквами $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}$ и т. д., чтобы отличать их от пропозициональных переменных. Формула алгебры логики формально определяется следующим образом:

1. Пропозициональная переменная — (атомарная) формула.
2. Если $\mathfrak{A}$ и $\mathfrak{B}$ — формулы, то $\mathfrak{A} \,\&\, \mathfrak{B}$, $\mathfrak{A} \vee \mathfrak{B}$, $\mathfrak{A} \to \mathfrak{B}$, $\mathfrak{A} \leftrightarrow \mathfrak{B}$ — формулы.
3. Если $\mathfrak{A}$ — формула, то $\overline{\mathfrak{A}}$ — формула.
4. Никаких других формул не существует.

[17] Ян Лукасе́вич (Jan Łukasiewicz; 1878—1956) — польский логик.
[18] от *араб.* الجبر — воссоединение, связь, завершение.
[19] от *лат.* formula — форма, правило, предписание.

Например, выражение $\big((X \,\&\, Y) \leftrightarrow \overline{Z}\big)$ является формулой.

Для формирования сложных формул используют вспомогательные символы — скобки «(» и «)». Лишние наружные скобки, как правило, опускают. К примеру, формулу $\big(A \,\&\, (B \vee C)\big)$ пишут как $A \,\&\, (B \vee C)$, а $(A \,\&\, B)$ как $A \,\&\, B$. Также вводится *приоритет операций:* операция отрицания «$\overline{\ }$» имеет наивысший приоритет, затем в порядке понижения приоритета идут «&», «$\vee$», «$\to$», «$\leftrightarrow$». Например, фомулу $\Big(A \vee \big(D \,\&\, (\overline{B})\big)\Big) \to (C \leftrightarrow E)$ можно записать так: $A \vee D \,\&\, \overline{B} \to (C \leftrightarrow E)$.

1.5. Булевы функции

Истинностное значение высказывания, полученного при помощи логических операций из более простых высказываний, полностью определяется истинностными значениями этих исходных высказываний. Поэтому каждой логической операции соответствует функция, принимающая значения из множества $\{0, 1\}$, аргументы которой также принимают значения из множества $\{0, 1\}$. Такие функции называют *булевыми*[20] *функциями,* или *логическими функциями,* или *функциями алгебры логики.*

Каждая булева функция может быть изображена посредством *таблицы истинности,* как в табл. 1.1.

Таблица 1.1

Таблица истинности для основных логических операций

A	B	$A \,\&\, B$	$A \vee B$	$A \to B$	$A \leftrightarrow B$	$\overline{A}$
0	0	0	0	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	0	0
1	1	1	1	1	1	0

Это так называемый *табличный способ задания* булевых функций. Наряду с ним применяется *задание функций с помощью*

[20] по имени основателя алгебры логики — английского математика и логика Джорджа Буля (George Boole; 1815—1864).

формул. Пусть $X_1, \ldots, X_n$ — некоторый фиксированный список переменных. Каждой формуле $\mathfrak{A}$, содержащей переменные только из этого списка, соответствует n-местная функция $f(X_1, \ldots, X_n)$. *Двоичным набором* будем называть упорядоченный набор значений 0 и 1. Пусть $\sigma_1, \ldots, \sigma_n$ — некоторый двоичный набор. Если переменные $X_1, \ldots, X_n$ имеют истинностные значения $\sigma_1, \ldots, \sigma_n$ соответственно, то, произведя вычисления в соответствии с операциями, указанными в формуле $\mathfrak{A}$, можно получить значение 0 или 1, которое и считается значением функции f на двоичном наборе $\sigma_1, \ldots, \sigma_n$. Говорят, что формула $\mathfrak{A}$ *реализует* функцию f. Например, формула $(X \,\&\, Y) \leftrightarrow \overline{Z}$ реализует функцию $h(X, Y, Z)$, изображённую в табл. 1.2.

Таблица 1.2

Таблица истинности для функции $h(X, Y, Z) = (X \,\&\, Y) \leftrightarrow \overline{Z}$

X	Y	Z	$h(X, Y, Z)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Очевидно, что для n-местной булевой функции таблица истинности будет содержать 2^n строк значений. Это делает табличный способ неприемлемым для задания функций с большим количеством аргументов.

1.6. Равносильные формулы

Пусть $\mathfrak{A}$ и $\mathfrak{B}$ — две формулы, содержащие лишь переменные из списка $X_1, \ldots, X_n$, и пусть $f(X_1, \ldots, X_n)$ и $g(X_1, \ldots, X_n)$ — функции, реализуемые формулами $\mathfrak{A}$ и $\mathfrak{B}$ соответственно. Формулы $\mathfrak{A}$ и $\mathfrak{B}$ называют *равносильными,* если функции f и g совпадают, т. е. совпадают их таблицы истинности. Для равносильных формул используется обозначение $\mathfrak{A} \equiv \mathfrak{B}$.

В алгебре логики важную роль играют следующие равенства:

1. Законы коммутативности[21)] (переместительности):

   а) $A \,\&\, B \equiv B \,\&\, A$ — коммутативность конъюнкции;

   б) $A \vee B \equiv B \vee A$ — коммутативность дизъюнкции.

[21)] от *лат.* commutativus — меняющийся, подвергающийся перемещению.

2. Законы ассоциативности[22] (сочетательности):

   а) $(A \& B) \& C \equiv A \& (B \& C)$ — ассоциативность конъюнкции;

   б) $(A \vee B) \vee C \equiv A \vee (B \vee C)$ — ассоциативность дизъюнкции.

3. Законы идемпотентности[23]:

   а) $A \& A \equiv A$ — идемпотентность конъюнкции;

   б) $A \vee A \equiv A$ — идемпотентность дизъюнкции.

4. Законы дистрибутивности[24] (распределительности):

   а) $A \& (B \vee C) \equiv (A \& B) \vee (A \& C)$ — дистрибутивность конъюнкции относительно дизъюнкции;

   б) $A \vee (B \& C) \equiv (A \vee B) \& (A \vee C)$ — дистрибутивность дизъюнкции относительно конъюнкции.

5. Свойства нуля и единицы:

   а) $A \& 0 \equiv 0$;

   б) $A \vee 0 \equiv A$;

   в) $A \& 1 \equiv A$;

   г) $A \vee 1 \equiv 1$.

6. $\overline{\overline{A}} \equiv A$ — правило снятия двойного отрицания.

7. Законы де Моргана[25]:

   а) $\overline{A \& B} \equiv \overline{A} \vee \overline{B}$;

   б) $\overline{A \vee B} \equiv \overline{A} \& \overline{B}$.

8. $A \& \overline{A} \equiv 0$ — закон противоречия.

9. $A \vee \overline{A} \equiv 1$ — закон исключённого третьего.

---

[22] от *лат.* associatio — соединение.

[23] от *лат.* idempotens — сохраняющий ту же степень. По законам идемпотентности в логике нет коэффициентов и показателей степени.

[24] от *лат.* distributio — размещение, распределение.

[25] Огастес (Август) де Мо́рган (Augustus de Morgan; 1806—1871) — шотландский математик и логик.

10. Законы поглощения:

    а) $A \,\&\, (A \vee B) \equiv A$;

    б) $A \vee (A \,\&\, B) \equiv A$.

11. Правила склеивания-расщепления[26]:

    а) $(A \vee \overline{B}) \,\&\, (A \vee B) \equiv A$;

    б) $(A \,\&\, \overline{B}) \vee (A \,\&\, B) \equiv A$.

12. $A \to B \equiv \overline{A} \vee B$ — выражение импликации через дизъюнкцию и отрицание.

13. $A \leftrightarrow B \equiv (A \to B) \,\&\, (B \to A)$ — выражение эквивалентности через конъюнкцию и импликацию.

14. Выражение эквивалентности через конъюнкцию, дизъюнкцию и отрицание:

    а) $A \leftrightarrow B \equiv (A \,\&\, B) \vee (\overline{A} \,\&\, \overline{B})$;

    б) $A \leftrightarrow B \equiv (\overline{A} \vee B) \,\&\, (A \vee \overline{B})$.

Из определения равенства формул можно извлечь следующие правила, позволяющие получать новые равносильные формулы из уже имеющихся:

1. *Правило подстановки.* Пусть $\mathfrak{A} \equiv \mathfrak{B}$, пусть $X_1, \ldots, X_n$ — все переменные, фигурирующие в формулах $\mathfrak{A}$ и $\mathfrak{B}$, и пусть формула $\mathfrak{A}'$ получается из $\mathfrak{A}$, а формула $\mathfrak{B}'$ — из $\mathfrak{B}$ путём подстановки некоторых формул $\mathfrak{C}_1, \ldots, \mathfrak{C}_n$ вместо переменных $X_1, \ldots, X_n$. Тогда $\mathfrak{A}' \equiv \mathfrak{B}'$. В более узком смысле это правило звучит так: *если в равносильных формулах вместо всех вхождений некоторой переменной подставить одну и ту же формулу, то получатся новые равносильные формулы.*

2. *Правило замены.* Пусть $\mathfrak{A} \equiv \mathfrak{B}$, а формула $\mathfrak{C}$ содержит $\mathfrak{A}$ в качестве составной части (т. е. $\mathfrak{A}$ является *подформулой* формулы $\mathfrak{C}$). Пусть $\mathfrak{C}'$ получается из $\mathfrak{C}$ подстановкой формулы $\mathfrak{B}$

[26] Если данные правила применяются для уменьшения числа операций, то говорят, что производится *склеивание,* а если наоборот, то *расщепление.*

вместо $\mathfrak{A}$. Тогда $\mathfrak{C}' \equiv \mathfrak{C}$. Иначе говоря, *если в формуле заменить некоторую подформулу на равносильную, то получится новая формула, равносильная исходной.*

Напомним, что в математике преобразование формулы в равносильную называется *эквивалентным преобразованием.*

1.7. Проблема разрешимости

Пусть задана некоторая формула $\mathfrak{A}$, реализующая булеву функцию $f(X_1, \ldots, X_n)$. Некоторый двоичный набор $\sigma_1, \ldots, \sigma_n$, который соответствует значениям переменных $X_1, \ldots, X_n$, будем называть *интерпретацией*[27] формулы $\mathfrak{A}$.

Формула $\mathfrak{A}$ называется *общезначимой,* или *тождественно истинной,* или *тавтологией*[28], если $\mathfrak{A} \equiv 1$. Очевидно, что в этом случае $f = 1$ на любой интерпретации.

Для общезначимой формулы также используется символическая запись с использованием знака выводимости: $\vdash \mathfrak{A}$.

Формула $\mathfrak{A}$ называется *невыполнимой,* или *тождественно ложной,* или *противоречием,* если $\mathfrak{A} \equiv 0$. Очевидно, что в этом случае $f = 0$ на любой интерпретации.

Формула $\mathfrak{A}$, не являющаяся ни противоречием, ни тавтологией, называется *выполнимой.* Для доказательства того, что формула $\mathfrak{A}$ является выполнимой, необходимо и достаточно найти такие интерпретации $\sigma_1, \ldots, \sigma_n$ и $\tau_1, \ldots, \tau_n$, что $f(\sigma_1, \ldots, \sigma_n) = 0$ и $f(\tau_1, \ldots, \tau_n) = 1$.

Пусть дана произвольная формула $\mathfrak{A}$. Можно ли как-то проверить, что она является общезначимой? Если существует такой способ (алгоритм), позволяющий за конечное число шагов убедиться в этом, то говорят, что проблема проверки общезначимости формул *(проблема разрешимости)* алгебры логики разрешима.

Теорема 1. *Проблема разрешимости в алгебре логики имеет положительное решение.*

Доказательство. Так как формуле $\mathfrak{A}$ однозначно соответствует n-местная булева функция $f(X_1, \ldots, X_n)$, причём n — конечное число, то имеем 2^n различных двоичных наборов $\sigma_1, \ldots, \sigma_n$.

[27] от *лат.* interpretatio — разъяснение, истолкование.

[28] от *греч.* τὸ αὐτό — то же самое и λόγος — слово, изречение.

Для каждого набора можно вычислить значение функции f за конечное число шагов. Найдя значения функции, мы узнаем, все ли они равны 1, т. е. истинна ли формула $\mathfrak{A}$ на всех интерпретациях. Если это так, то $\vdash \mathfrak{A}$. Таким образом, мы можем проверить общезначимость любой формулы за конечное число шагов. Теорема 1 доказана.

1.8. Релейно-контактные схемы

Каждой булевой функции можно поставить в соответствие так называемую *релейно-контактную схему*[29].

Релейно-контактная схема строится в предположении, что пропозициональная переменная X — это *замыкающий контакт* в электрической схеме, который замкнут при подаче (управляющего) тока X, т. е. $X = 1$, и разомкнут при его отсутствии — $X = 0$. Далее, формуле $\overline{X}$ отвечает *размыкающий контакт,* который замкнут, т. е. $\overline{X} = 1$, пока нет тока ($X = 0$), и размыкается — $\overline{X} = 0$, когда ток есть ($X = 1$).

Конъюнкции $X \,\&\, Y$ соответствует последовательное соединение (замыкающих) контактов, а дизъюнкции $X \vee Y$ — параллельное соединение (см. рис. 1.1).

а) $f(X, Y) = X \,\&\, Y$ *б)* $f(X, Y) = X \vee Y$

Рис. 1.1. Простейшие релейно-контактные схемы

Из простейших схем легко собираются произвольные релейно-контактные схемы. Например, на рис. 1.2 изображена релейно-кон-

[29] от *англ.* relay — смена. Австрийский и нидерландский физик-теоретик Пауль Э́ренфест (Paul Ehrenfest; 1880—1933) в 1910 г. первым высказал гипотезу о возможности применения алгебры логики в технике. Использовать реле для построения логических схем предложил в 1937 г. Клод Элвуд Ше́ннон (Claude Elwood Shannon; 1916—2001) — американский математик и инженер.

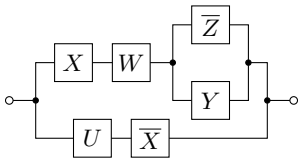

Рис. 1.2. Релейно-контактная схема для сложной булевой функции

тактная схема для булевой функции

$$f(U,W,X,Y,Z) = \big(X \,\&\, W \,\&\, (\overline{Z} \vee Y)\big) \vee (U \,\&\, \overline{X}).$$

1.9. Нормальные формы

Особую роль в алгебре логики и её приложениях играют так называемые дизъюнктивные и конъюнктивные нормальные формы, определяемые следующим образом.

Введём следующие обозначения. Пропозициональную переменную с отрицанием или без него будем называть *литералом*[30]:

$$X^\sigma = \begin{cases} X & \text{при } \sigma = 1, \\ \overline{X} & \text{при } \sigma = 0. \end{cases}$$

Формула $X_1^{\sigma_1} \,\&\, \ldots \,\&\, X_n^{\sigma_n}$, где $\sigma_1, \ldots, \sigma_n$ — какой-либо двоичный набор, а среди переменных могут быть одинаковые, называется *элементарной конъюнкцией* или *конъюнктом.*

Всякая дизъюнкция элементарных конъюнкций называется *дизъюнктивной нормальной формой (ДНФ).*

Аналогично, формула вида $X_1^{\sigma_1} \vee \ldots \vee X_n^{\sigma_n}$ называется *элементарной дизъюнкцией* или *дизъюнктом*, а конъюнкция элементарных дизъюнкций называется *конъюнктивной нормальной формой (КНФ).*

С помощью эквивалентных преобразований из § 1.6 для всякой формулы можно построить равносильные ей ДНФ и КНФ. Это делается следующим образом.

[30] от *лат.* littera — буква.

1. С помощью равенств 12 и 14 удалить все знаки импликации и эквивалентности.

2. Применяя равенства 6 и 7(а, б) можно добиться того, чтобы знак отрицания стоял только над переменными и чтобы не было многократных отрицаний.

3. С помощью законов дистрибутивности 4(а, б) получить КНФ или ДНФ.

Совершенной ДНФ (СДНФ) называют ДНФ, в которой нет равных элементарных конъюнкций и все элементарные конъюнкции содержат одни и те же переменные, причём каждую переменную — только один раз (включая её вхождение под знаком отрицания). Аналогично определяется *совершенная КНФ (СКНФ),* как такая КНФ, в которой нет одинаковых дизъюнктов; все дизъюнкты содержат одни и те же переменные, причём каждую — только один раз (включая отрицания).

Для каждой функции алгебры логики $f(X_1, \ldots, X_n)$, не равной тождественно 0, можно построить реализующую её СДНФ, причём единственную:

$$f(X_1, \ldots, X_n) = \bigvee_{f(\sigma_1, \ldots, \sigma_n)=1} X_1^{\sigma_1} \,\&\, \ldots \,\&\, X_n^{\sigma_n},$$

где дизъюнкция берётся по тем двоичным наборам $\sigma_1, \ldots, \sigma_n$, на которых функция f равна 1. Иными словами, сначала в таблице истинности отыскиваются строки, в которых функция f равна 1; каждой такой строке соответствует конъюнкт, состоящий из переменных, причём переменная входит в него под отрицанием, если в рассматриваемой строке значение данной переменной равно 0, и без отрицания, если её значение равно 1.

Например, функция $h(X, Y, Z)$, изображённая в табл. 1.2, имеет такую СДНФ:

$$(\overline{X} \,\&\, \overline{Y} \,\&\, Z) \vee (\overline{X} \,\&\, Y \,\&\, Z) \vee (X \,\&\, \overline{Y} \,\&\, Z) \vee (X \,\&\, Y \,\&\, \overline{Z}).$$

Каждая функция алгебры логики $f(X_1, \ldots, X_n)$, не равная тождественно 1, реализуется следующей СКНФ, причём единственной:

$$f(X_1, \ldots, X_n) = \bigwedge_{f(\sigma_1, \ldots, \sigma_n)=0} \overline{X_1^{\sigma_1}} \vee \ldots \vee \overline{X_n^{\sigma_n}},$$

где конъюнкция берётся по тем двоичным наборам $\sigma_1, \ldots, \sigma_n$, на которых функция f равна 0. Проще говоря, сначала в таблице истинности отыскиваются строки, в которых функция f равна 0; каждой такой строке соответствует дизъюнкт, состоящий из переменных, причём переменная входит в него под отрицанием, если в рассматриваемой строке значение данной переменной равно 1, и без отрицания, если её значение равно 0.

Так, СКНФ для той же функции $h(X, Y, Z)$ имеет вид

$$(X \vee Y \vee Z) \,\&\, (X \vee \overline{Y} \vee Z) \,\&\, (\overline{X} \vee Y \vee Z) \,\&\, (\overline{X} \vee \overline{Y} \vee \overline{Z}).$$

Вышеописанные методы нахождения СДНФ и СКНФ предполагают табличный способ задания функции. Рассмотрим другой метод получения совершенных нормальных форм, который основан на эквивалентных преобразованиях:

1. Для формулы $\mathfrak{A}$ получить любую ДНФ (КНФ).
2. Если в ДНФ (КНФ) есть конъюнкт (дизъюнкт) $\mathfrak{B}$, не содержащий переменную X_i, то необходимо её добавить, используя правила расщепления: $\mathfrak{B} \equiv (\mathfrak{B} \,\&\, X_i) \vee (\mathfrak{B} \,\&\, \overline{X_i})$ — для ДНФ, $\mathfrak{B} \equiv (\mathfrak{B} \vee X_i) \,\&\, (\mathfrak{B} \vee \overline{X_i})$ — для КНФ.
3. Если в ДНФ (КНФ) встречаются равные конъюнкты (дизъюнкты), то лишние нужно отбросить.
4. Если в ДНФ (КНФ) в некотором конъюнкте (дизъюнкте) литерал X_i встречается несколько раз, то лишние нужно отбросить.

   Если в ДНФ (КНФ) в некотором конъюнкте (дизъюнкте) литерал $\overline{X_i}$ встречается несколько раз, то лишние нужно отбросить.
5. Если в ДНФ некоторый конъюнкт содержит конъюнкцию $X_i \,\&\, \overline{X_i}$, то данный конъюнкт нужно отбросить.

   Если в КНФ некоторый дизъюнкт содержит дизъюнкцию $X_i \vee \overline{X_i}$, то данный дизъюнкт нужно отбросить.

1.10. Принцип двойственности

Функция алгебры логики f^* называется *двойственной* для функции f, если таблицу истинности для f^* можно получить из таблицы для f, заменив в ней всюду 1 на 0 и 0 на 1, т. е. функция

$f^*(X_1,\dots,X_n)$, двойственная к функции $f(X_1,\dots,X_n)$, удовлетворяет равенству

$$f^*(X_1,\dots,X_n)=\overline{f(\overline{X_1},\dots,\overline{X_n})}.$$

Например, конъюнкция и дизъюнкция двойственны друг другу.

Функция, совпадающая со своей двойственной, называется *самодвойственной.* Самодвойственная функция на противоположных наборах $\sigma_1,\dots,\sigma_n$ и $\overline{\sigma_1},\dots,\overline{\sigma_n}$ принимает противоположные значения. Примером самодвойственной функции является отрицание.

Если в формуле $\mathfrak{A}$ заменить знаки всех логических функций на знаки двойственных функций, то получится *двойственная формула* $\mathfrak{A}^*$, реализующая функцию, двойственную той, которая реализуется формулой $\mathfrak{A}$. Например, если некоторая формула алгебры логики содержит только операции $\&,\vee,\bar{\ }$ (не содержит операций $\to$ и $\leftrightarrow$), то получить двойственную к ней можно заменой $\&,\vee,1,0$ соответственно на $\vee,\&,0,1$.

Принцип двойственности заключается в следующем. Если $\mathfrak{A}\equiv\mathfrak{B}$, то верно и равенство $\mathfrak{A}^*\equiv\mathfrak{B}^*$, называемое *двойственным* предыдущему. Другими словами, если две формулы равносильны, то двойственные им формулы тоже равносильны. Например, двойственны друг другу равенства 1(а) и 1(б), 2(а) и 2(б), 3(а) и 3(б), 4(а) и 4(б), 5(а) и 5(г), 5(б) и 5(в), 7(а) и 7(б), 8 и 9, 10(а) и 10(б), 11(а) и 11(б) из § 1.6. Каждая КНФ имеет двойственную ей ДНФ.

1.11. Логическое следствие

Логика — наука о способах доказательства. В булевой алгебре все доказательства строились на *отношении эквивалентности.* Две логические функции считались эквивалентными, если на одинаковых двоичных наборах они выдавали одинаковые значения. В логике высказываний все доказательства строятся на *отношении порядка,* т. е. на отношении, которое существует между причиной и следствием. Здесь уже отдельные звенья цепи доказательства связаны с помощью импликации.

Пусть даны формулы $\mathfrak{A}_1,\mathfrak{A}_2,\dots,\mathfrak{A}_m,\mathfrak{B}$. Формула $\mathfrak{B}$ является *логическим следствием (логическим следованием)* формул $\mathfrak{A}_1,\mathfrak{A}_2,\dots,\mathfrak{A}_m$, если, придавая значения переменным $X_1,X_2,\dots,X_n$,

от которых зависят все рассматриваемые формулы, всякий раз, когда истинны одновременно все формулы $\mathfrak{A}_1, \mathfrak{A}_2, \ldots, \mathfrak{A}_m$, истинна и формула $\mathfrak{B}$.

Для логического следствия используется запись

$$\mathfrak{A}_1, \mathfrak{A}_2, \ldots, \mathfrak{A}_m \vdash \mathfrak{B}.$$

Слева от знака «$\vdash$» располагаются причины (посылки), справа — следствия (заключения).

Форму мышления, при которой новая мысль выводятся по правилам логики из некоторых данных мыслей-посылок, называют *дедукцией*[31].

Т е о р е м а 2 (о дедукции). $\mathfrak{A} \vdash \mathfrak{B}$ *тогда и только тогда, когда* $\vdash (\mathfrak{A} \to \mathfrak{B})$.

Следующее утверждение является важным следствием теоремы о дедукции. Оно иллюстрирует связь между отношением логического следствия и операцией импликации.

С л е д с т в и е (из теоремы 2).

$$\mathfrak{A}_1, \mathfrak{A}_2, \ldots, \mathfrak{A}_m \vdash \mathfrak{B}$$

тогда и только тогда, когда

$$\vdash (\mathfrak{A}_1 \,\&\, \mathfrak{A}_2 \,\&\, \ldots \,\&\, \mathfrak{A}_m) \to \mathfrak{B}.$$

1.12. Силлогизмы

Силлогизм[32] — это правило, основанное на отношении порядка, позволяющее из истинных высказываний получать новые истинные высказывания. Приведём неполный список силлогизмов:

1. $\dfrac{A,\ A \to B}{B}$ — modus ponens[33].
2. $\dfrac{A \to B,\ \overline{B}}{\overline{A}}$ — modus tollens[34].

[31] от *лат.* deductio — выведение.
[32] от *греч.* συλλογισμός — сосчитывание.
[33] Утверждающий способ рассуждения *(лат.)*.
[34] Отрицающий способ рассуждения *(лат.)*.

3. $\dfrac{A \to B,\ B \to C}{A \to C}$ — цепное заключение.

4. $\dfrac{A,\ B}{A \,\&\, B}$ — введение конъюнкции.

5. $\dfrac{A \,\&\, B}{A}$; $\dfrac{A \,\&\, B}{B}$ — удаление конъюнкции.

6. $\dfrac{A \vee B,\ \overline{B}}{A}$ — modus tollendo ponens[35)].

7. $\dfrac{\overline{A \,\&\, B},\ A}{\overline{B}}$ — modus ponendo tollens[36)].

8. $\dfrac{A \to (B \to C)}{B \to (A \to C)}$ — перестановка посылок.

9. $\dfrac{A \to (B \to C)}{(A \,\&\, B) \to C}$ — соединение посылок.

10. $\dfrac{(A \,\&\, B) \to C}{A \to (B \to C)}$ — разъединение посылок.

Формула, расположенная в нижней части силлогизма (под чертой), является логическим следствием формул, расположенных в верхней части силлогизма (над чертой).

1.13. Примеры решения задач

Пример 1.1. Записать символически следующее утверждение: *Если «Динамо» проиграет свой последний матч, то в случае выигрыша «Спартака» он («Спартак») станет чемпионом.*

Р е ш е н и е. Введём следующие обозначения:

W — «„Динамо“ выиграет свой последний матч»;

S — «„Спартак“ выиграет»;

C — «„Спартак“ станет чемпионом».

[35)] Отрицающее-утверждающий способ рассуждения *(лат.)*.
[36)] Утверждающе-отрицающий способ рассуждения *(лат.)*.

Исходное высказывание получаем из элементарных с помощью пропозициональных связок: $\overline{W} \to (S \to C)$.

О т в е т: $\overline{W} \to (S \to C)$.

Пример 1.2. Привести формулу $(A \to B) \to C$ к совершенному конъюнктивному нормальному виду.

Р е ш е н и е. При помощи эквивалентных преобразований получим сначала конъюнктивную нормальную форму данной формулы:

$$(A \to B) \to C \equiv (\overline{A} \vee B) \to C \equiv \overline{(\overline{A} \vee B)} \vee C \equiv$$
$$\equiv (\overline{\overline{A}} \,\&\, \overline{B}) \vee C \equiv (A \,\&\, \overline{B}) \vee C \equiv (A \vee C) \,\&\, (\overline{B} \vee C).$$

Теперь формула находится в КНФ, но это не совершенный её вид (в первой скобке отсутствует литерал B, а во второй — литерал A). Чтобы привести формулу к СКНФ, выполним следующие преобразования:

$$(A \vee C) \,\&\, (\overline{B} \vee C) \equiv (A \vee C \vee B) \,\&\, (A \vee C \vee \overline{B}) \,\&\, (\overline{B} \vee C) \equiv$$
$$\equiv (A \vee C \vee B) \,\&\, (A \vee C \vee \overline{B}) \,\&\, (\overline{B} \vee C \vee A) \,\&\, (\overline{B} \vee C \vee \overline{A}) \equiv$$
$$\equiv (A \vee B \vee C) \,\&\, (A \vee \overline{B} \vee C) \,\&\, (\overline{A} \vee \overline{B} \vee C).$$

О т в е т: $(A \vee B \vee C) \,\&\, (A \vee \overline{B} \vee C) \,\&\, (\overline{A} \vee \overline{B} \vee C)$.

Пример 1.3. При каких значениях пропозициональных переменных формула $(A \to B) \to C$ принимает ложные значения?

Р е ш е н и е. Приведём данную формулу к СКНФ (см. предыдущий пример): $(A \vee B \vee C) \,\&\, (A \vee \overline{B} \vee C) \,\&\, (\overline{A} \vee \overline{B} \vee C)$. Чтобы формула приняла ложное значение, достаточно, чтобы хотя бы один из дизъюнктов её СКНФ принял ложное значение.

Рассмотрим один из дизъюнктов: $(A \vee \overline{B} \vee C)$. Чтобы дизъюнкт принял ложное значение, необходимо, чтобы все составляющие его литералы приняли ложное значение. Данный дизъюнкт примет ложное значение на интерпретации $\{A = 0,\ B = 1,\ C = 0\}$.

Итого, исходная формула принимает ложное значение на трёх интерпретациях (соответственно дизъюнктам её СКНФ):

$$\{A = 0, B = 0, C = 0\},\ \{A = 0, B = 1, C = 0\},\ \{A = 1, B = 1, C = 0\}.$$

О т в е т: данная формула принимает ложное значение на следующих интерпретациях:

$$\{A=0,B=0,C=0\},\ \{A=0,B=1,C=0\},\ \{A=1,B=1,C=0\}.$$

Пример 1.4. Доказать, что если формулы $\overline{A\,\&\,B}$ и A принимают истинное значение, то и формула $\overline{B}$ также принимает истинное значение.

Д о к а з а т е л ь с т в о. Задачу можно переформулировать следующим образом: *доказать, что формула* $\overline{B}$ *является логическим следствием формул* $\overline{A\,\&\,B}$ *и* A. Для этого докажем, что формула $\overline{A\,\&\,B}\,\&\,A\to\overline{B}$ является тождественно истинной (см. § 1.11):

$$\overline{A\,\&\,B}\,\&\,A\to\overline{B}\equiv\overline{\overline{A\,\&\,B}\,\&\,A}\vee\overline{B}\equiv(A\,\&\,B)\vee(\overline{A}\vee\overline{B})\equiv$$
$$\equiv(\overline{A}\vee\overline{B}\vee A)\,\&\,(\overline{A}\vee\overline{B}\vee B)\equiv(\overline{B}\vee 1)\,\&\,(\overline{A}\vee 1)\equiv 1.$$

Итак, формула $\overline{A\,\&\,B}\,\&\,A\to\overline{B}$ является тождественно истинной, а значит формула $\overline{B}$ является логическим следствием формул $\overline{A\,\&\,B}$ и A, что и требовалось доказать.

Пример 1.5. Проверить правильность рассуждения: *Алексей старше Бориса или Борис старше Виктора. Однако, Борис не старше Виктора. Следовательно, Алексей старше Бориса.*

Таблица 1.3

Таблица истинности для примера 1.5

A	B	$A\vee B$	$\overline{B}$	A
0	0	0	1	0
0	1	1	0	0
1	0	**1**	**1**	**1**
1	1	1	0	1

Р е ш е н и е. Введём следующие обозначения:

A — «Алексей старше Бориса»;

B — «Борис старше Виктора».

Исходную задачу можем переформулировать следующим образом: *верно ли, что формула* A *является логическим следствием формул* $A\vee B$ *и* $\overline{B}$*?* Построим таблицу истинности для этих формул (табл. 1.3).

Из выделенной строки таблицы видно, что формула A действительно является логическим следствием формул $A\vee B$ и $\overline{B}$ (следствие истинно, когда истинны одновременно все посылки; см. § 1.11).

О т в е т: данное рассуждение верно.

Пример 1.6. Найти все не равносильные между собой и не тождественно истинные формулы алгебры высказываний, являющиеся логическими следствиями следующих формул (посылок):

$$X \to (Y \vee Z), \quad Z \to Y.$$

Р е ш е н и е. Составим конъюнкцию посылок и эквивалентными преобразованиями приведём её к СКНФ:

$$\big(X \to (Y \vee Z)\big) \,\&\, (Z \to Y) \equiv (\overline{X} \vee Y \vee Z) \,\&\, (\overline{Z} \vee Y) \equiv$$
$$\equiv (\overline{X} \vee Y \vee Z) \,\&\, (X \vee Y \vee \overline{Z}) \,\&\, (\overline{X} \vee Y \vee \overline{Z}).$$

Логическими следствиями из данных посылок будут все дизъюнкты, входящие в полученную СКНФ, а также всевозможные конъюнкции этих дизъюнктов по два, по три и т. д. Выпишем получающиеся формулы и упростим их:

1) $\overline{X} \vee Y \vee Z \equiv X \to (Y \vee Z)$ — первая посылка;

2) $X \vee Y \vee \overline{Z} \equiv Z \to (X \vee Z)$;

3) $\overline{X} \vee Y \vee \overline{Z} \equiv (X \,\&\, Z) \to Y$;

4) $(\overline{X} \vee Y \vee Z) \,\&\, (X \vee Y \vee \overline{Z}) \equiv \big((\overline{X} \vee Z) \,\&\, (X \vee \overline{Z})\big) \vee Y \equiv (X \leftrightarrow Z) \vee Y$;

5) $(\overline{X} \vee Y \vee Z) \,\&\, (\overline{X} \vee Y \vee \overline{Z}) \equiv \overline{X} \vee Y \equiv X \to Y$;

6) $(X \vee Y \vee \overline{Z}) \,\&\, (\overline{X} \vee Y \vee \overline{Z}) \equiv Y \vee \overline{Z} \equiv Z \to Y$ — вторая посылка;

7) $(\overline{X} \vee Y \vee Z) \,\&\, (X \vee Y \vee \overline{Z}) \,\&\, (\overline{X} \vee Y \vee \overline{Z}) \equiv (X \vee Z) \to Y$.

О т в е т: логическими следствиями указанных посылок являются следующие формулы: $X \to (Y \vee Z)$, $Z \to (X \vee Z)$, $(X \,\&\, Z) \to Y$, $(X \leftrightarrow Z) \vee Y$, $X \to Y$, $Z \to Y$, $(X \vee Z) \to Y$.

Глава 2

Логика предикатов Метод резолюций

2.1. Предикаты и формулы

Логика предикатов — это расширение возможностей логики высказываний, позволяющее строить высказывания с учётом свойств изучаемых объектов или отношений между ними.

Одноместный предикат[1] $P(x)$ — это утверждение об объекте x, где x рассматривается как переменная. Иначе говоря, если в $P(x)$ вместо x подставить конкретный изучаемый объект a, то получаем высказывание, принадлежащее алгебре высказываний. В таком случае $P(a) = 1$ или $P(a) = 0$.

Предикаты будем обозначать заглавными латинскими буквами, переменные — строчными буквами из конца латинского алфавита $(x, y, z, \ldots)$, константы — строчными буквами из начала латинского алфавита $(a, b, c, \ldots)$.

Многоместный предикат $P(x_1, \ldots, x_n)$ — это утверждение об объектах $x_1, \ldots, x_n$, где $x_1, \ldots, x_n$ рассматриваются как переменные. Следовательно, при подстановке конкретных значений $a_1, \ldots, a_n$ получим высказывание $P(a_1, \ldots, a_n)$, являющееся истинным или ложным.

Для получения высказываний из предикатов помимо подстановки вместо переменных конкретных объектов применяются также *кванторы*[2]: *квантор всеобщности* «$\forall$» и *квантор существования* «$\exists$».

Символ «$\forall x$» интерпретируется как фраза «для всех x»[3]. При добавлении к предикату $P(x)$ квантора всеобщности мы получим высказывание $\forall x\, P(x)$. Оно примет истинное значение, если предикат $P(x)$ выполняется *для всех* объектов $a_1, a_2, \ldots, a_i, \ldots$, которые

---

[1] от *лат.* prædicatum — сказанное.

[2] от *лат.* quantum — сколько. Кванторы показывают, о скольких объектах говорится в предложении.

[3] Знак «$\forall$» произошёл от первой буквы английского слова all — «все».

можно подставить вместо x. Это можно записать как

$$\forall x\, P(x) \equiv P(a_1)\ \&\ P(a_2)\ \&\ \ldots\ \&\ P(a_i)\ \&\ \ldots$$

Символ «$\exists x$» представляет фразу «существует x»[4]. При добавлении данного квантора к предикату получаем также высказывание $\exists x\, P(x)$. Данное высказывание будет истинным, если предикат $P(x)$ выполняется *хотя бы для одного* из объектов $a_1, a_2, \ldots, a_i, \ldots$, которые можно подставить вместо x. Это можно записать как

$$\exists x\, P(x) \equiv P(a_1) \vee P(a_2) \vee \ldots \vee P(a_i) \vee \ldots$$

Переменная x, входящая в формулу $\mathfrak{A}$, называется *связанной,* если она находится под действием квантора $\forall x$ или $\exists x$. В противном случае, переменная x в формуле $\mathfrak{A}$ является *свободной.* Например, в формулах $\exists x(x = y)$ и $\forall x\, B(x, y)$ переменная x связанная, а переменная y свободная. Очевидно, что формула без свободных переменных является высказыванием.

2.2. Интерпретации

За каждой формулой скрывается нечто, в связи с чем она была написана и о чём она повествует, то есть скрывается её содержание. Содержательная часть формул, их смысл, относится к специальному разделу логики, называемому *семантикой*[5]. Выяснить содержание формулы можно, обращаясь к реальному миру предметов. Делается это посредством интерпретации формулы.

Чтобы определить *интерпретацию* для формулы логики первого порядка, мы должны указать предметную область (область значений предметных переменных) и значения констант, функциональных и предикатных символов, встречающихся в формуле.

Интерпретация формулы $\mathfrak{A}$ логики предикатов состоит из непустой (предметной) области $\mathscr{D}$ и указания значения всех констант, функциональных символов и предикатных символов, встречающихся в $\mathfrak{A}$. То есть каждой константе мы ставим в соответствие некоторый элемент из $\mathscr{D}$, каждому n-местному функциональному символу

[4] Знак «∃» произошёл от первой буквы английского слова exists — «существует».

[5] от *греч.* σημαντικός — обозначающий.

$f(x_1, \ldots, x_n)$ мы ставим в соответствие отображение из $\mathscr{D}^n$ в $\mathscr{D}$:

$$\underbrace{\mathscr{D} \times \mathscr{D} \times \ldots \times \mathscr{D}}_{n \text{ раз}} \xrightarrow{f} \mathscr{D},$$

а каждому n-местному предикатному символу $P(x_1, \ldots, x_n)$ мы ставим в соответствие отображение из $\mathscr{D}^n$ в $\{0, 1\}$:

$$\underbrace{\mathscr{D} \times \mathscr{D} \times \ldots \times \mathscr{D}}_{n \text{ раз}} \xrightarrow{P} \{0, 1\}.$$

При интерпретации формулы наполняются содержанием благодаря тому, что элементы множества $\mathscr{D}$ уже привязаны к конкретной реальности, знакомы и понятны. Например, в случае, когда $\mathscr{D} = \mathbb{R}$, т. е. область $\mathscr{D}$ является множеством действительных чисел, у нас не появляется чувства беспокойства, под формулами мы понимаем сведения из математического анализа, который прочно привязан к практической деятельности инженеров, физиков, химиков и т. д. Следовательно, нам становится ясным, когда формула при определённой фиксации своих переменных истинна, а когда ложна.

2.3. Истинность формул. Логическое следствие

Формула $\mathfrak{A}$ логики предикатов называется *выполнимой в области $\mathscr{M}$,* если существуют значения переменных, входящих в эту формулу и отнесённых к области $\mathscr{M}$ (иначе — *существует модель*), при которых формула $\mathfrak{A}$ принимает истинные значения.

Формула $\mathfrak{A}$ логики предикатов называется *выполнимой,* если существует область, на которой эта формула выполнима.

Формула $\mathfrak{A}$ логики предикатов называется *тождественно истинной в области $\mathscr{M}$,* если она принимает истинные значения для всех значений переменных, входящих в эту формулу и отнесённых к этой области.

Формула $\mathfrak{A}$ логики предикатов называется *общезначимой (тавтологией),* если она является тождественно истинной на всякой области (на любой модели). Для общезначимой формулы пишем $\vdash \mathfrak{A}$. Если формула $\mathfrak{A}$ общезначимая, то формула $\overline{\mathfrak{A}}$ называется *тождественно ложной,* или *противоречием.*

Пусть даны две формулы $\mathfrak{A}(x_1,\ldots,x_n)$ и $\mathfrak{B}(x_1,\ldots,x_n)$. Формула $\mathfrak{B}$ является *логическим следствием* формулы $\mathfrak{A}$, если во всякой интерпретации формула $\mathfrak{B}$ выполнена на каждом наборе переменных $(x_1{=}a_1),\ \ldots,\ (x_n{=}a_n)$, на котором выполнена формула $\mathfrak{A}$. Символически для логического следствия используют обозначение $\mathfrak{A}\vdash\mathfrak{B}$.

В логике предикатов выполняется *теорема дедукции:* $\mathfrak{A}\vdash\mathfrak{B}$ тогда и только тогда, когда $\vdash\mathfrak{A}\to\mathfrak{B}$.

Формулы $\mathfrak{A}(x_1,\ldots,x_n)$ и $\mathfrak{B}(x_1,\ldots,x_n)$ называются равносильными, если $\mathfrak{A}\vdash\mathfrak{B}$ и $\mathfrak{B}\vdash\mathfrak{A}$. Для равносильных формул используется запись: $\mathfrak{A}\equiv\mathfrak{B}$.

Примеры равносильных формул:

$$
\begin{aligned}
\forall x\,\mathfrak{A}(x)\ \&\ \mathfrak{B} &\equiv \forall x\big(\mathfrak{A}(x)\ \&\ \mathfrak{B}\big);\\
\exists x\,\mathfrak{A}(x)\ \&\ \mathfrak{B} &\equiv \exists x\big(\mathfrak{A}(x)\ \&\ \mathfrak{B}\big);\\
\forall x\,\mathfrak{A}(x)\vee\mathfrak{B} &\equiv \forall x\big(\mathfrak{A}(x)\vee\mathfrak{B}\big);\\
\exists x\,\mathfrak{A}(x)\vee\mathfrak{B} &\equiv \exists x\big(\mathfrak{A}(x)\vee\mathfrak{B}\big);\\
\overline{\forall x\,\mathfrak{A}(x)} &\equiv \exists x\,\overline{\mathfrak{A}(x)};\\
\overline{\exists x\,\mathfrak{A}(x)} &\equiv \forall x\,\overline{\mathfrak{A}(x)};\\
\forall x\,\mathfrak{A}(x)\ \&\ \forall x\,\mathfrak{B}(x) &\equiv \forall x\big(\mathfrak{A}(x)\ \&\ \mathfrak{B}(x)\big);\\
\exists x\,\mathfrak{A}(x)\vee\exists x\,\mathfrak{B}(x) &\equiv \exists x\big(\mathfrak{A}(x)\vee\mathfrak{B}(x)\big);\\
\forall x\,\mathfrak{A}(x)\to\mathfrak{B} &\equiv \exists x\big(\mathfrak{A}(x)\to\mathfrak{B}\big);\\
\exists x\,\mathfrak{A}(x)\to\mathfrak{B} &\equiv \forall x\big(\mathfrak{A}(x)\to\mathfrak{B}\big);\\
\mathfrak{B}\to\forall x\,\mathfrak{A}(x) &\equiv \forall x\big(\mathfrak{B}\to\mathfrak{A}(x)\big);\\
\mathfrak{B}\to\exists x\,\mathfrak{A}(x) &\equiv \exists x\big(\mathfrak{B}\to\mathfrak{A}(x)\big).
\end{aligned}
$$

2.4. Сколемовские функции и сколемизация формул

Всякую формулу логики предикатов с помощью эквивалентных преобразований можно привести к равносильной формуле в *приведённой форме,* в которой из логических операций используются только операции &, $\vee$, $\overline{\ }$, причём отрицания относятся только к предикатным символам и элементарным высказываниям.

Далее, всякую формулу логики предикатов, находящуюся в приведённой форме, с помощью эквивалентных преобразований можно привести к равносильной формуле в *предварённой нормальной форме (ПНФ),* в которой все кванторы стоят в её начале, а область действия каждого из них распространяется до конца формулы, т. е. привести к виду

$$\text{Ч}_1 x_1 \dots \text{Ч}_n x_n \dots \ \mathfrak{B}(x_1, \dots, x_n, \dots),$$

где $\text{Ч}_1, \dots, \text{Ч}_n$ — кванторы (либо $\exists$, либо $\forall$), а формула $\mathfrak{B}$ не содержит кванторов. Такую формулу $\mathfrak{B}$ иногда называют *матрицей.* Заметим, что в ПНФ кванторов может и не быть вовсе.

Легко добиться того, чтобы последними стояли кванторы существования. Для этого используется равенство:

$$\mathfrak{B}(x_1, \dots, x_n, \dots) \ \equiv \ \exists u \big(\mathfrak{B}(x_1, \dots, x_n, \dots) \,\&\, \mathfrak{T}(u) \big),$$

где $\mathfrak{T}(u)$ — произвольная общезначимая формула.

Будем теперь *элиминировать*[6)] в формуле $\mathfrak{A}$ последовательно кванторы существования слева направо, заменяя их на функции. Путеводной здесь является идея, что пара кванторов $\forall u \exists v$ — это некоторая функция $v = f(u)$. Набору кванторов $\forall y_1 \dots \forall y_{n_2} \exists z_i$ отвечает функция $z_i = g_i(y_1, \dots, y_{n_2})$.

Если самой левой группой кванторов являются кванторы существования $\exists x_1 \dots \exists x_{n_1}$, то им соответствуют некоторые конкретные значения (константы) $a_1, \dots, a_{n_1}$.

Элиминируя группу кванторов в формуле $\mathfrak{A}$, мы в $\mathfrak{B}$ оставляем переменные $y_1, \dots, y_{n_2}$, по которым стояли кванторы всеобщности, а вместо следующих за ними переменных $z_1, \dots, z_{n_3}$, связанных кванторами существования, подставляем полученные функции $g_1(y_1, \dots, y_{n_2}), \dots, g_{n_3}(y_1, \dots, y_{n_2})$.

В результате имеем набор *сколемовских функций*

$$\begin{aligned} &a_1, \dots, a_{n_1}, \\ &g_1(y_1, \dots, y_{n_2}), \dots, g_{n_3}(y_1, \dots, y_{n_2}), \\ &\vdots \\ &h_1(w_1, \dots, w_{n_{k-1}}), \dots, h_{n_k}(w_1, \dots, w_{n_{k-1}}) \end{aligned}$$

[6)] от *лат.* eliminare — исключать, удалять.

и формулу

$$\begin{aligned}\mathfrak{A}_{\mathrm{S}} = \ & \forall y_1 \ldots \forall y_{n_2} \ \ldots \ \forall w_1 \ldots \forall w_{n_{k-1}} \mathfrak{B}\big(a_1, \ldots, a_{n_1}, \\ & y_1, \ldots, y_{n_2},\ g_1(y_1, \ldots, y_{n_2}), \ldots, g_{n_3}(y_1, \ldots, y_{n_2}), \\ & \vdots \\ & w_1, \ldots, w_{n_{k-1}},\ h_1(w_1, \ldots, w_{n_{k-1}}), \ldots, h_{n_k}(w_1, \ldots, w_{n_{k-1}})\big).\end{aligned}$$

Формула $\mathfrak{A}_{\mathrm{S}}$ не содержит кванторов существования и является *сколемовской нормальной формой* исходной формулы. Любую формулу логики предикатов можно привести к сколемовской нормальной форме с сохранением противоречивости. Из этого утверждения вытекает тот факт, что исходная формула и её сколемовская нормальная форма в общем случае не являются равносильными формулами.

Идея использования функций вместо групп кванторов восходит к работам Т. Сколема[7] и Ж. Эрбрана[8]. Поэтому такие функции называют сколемовскими или (реже) эрбрановскими, а их добавление — *сколемизацией.*

2.5. Метод резолюций

2.5.1. Метод резолюций в логике высказываний

Предположим, что дана формула логики высказываний $\mathfrak{A}$. Как проверить общезначимость формулы $\mathfrak{A}$? Эта задача решается с помощью *метода резолюций*[9], предложенного Дж. Робинсоном[10] в 1965 г. Очевидно, что $\vdash \mathfrak{A}$ тогда и только тогда, когда формула $\overline{\mathfrak{A}}$ является противоречием.

Формулу $\overline{\mathfrak{A}}$ приводим к КНФ, т. е. в данном случае

$$\overline{\mathfrak{A}} = \mathfrak{D}_1 \,\&\, \ldots \,\&\, \mathfrak{D}_p,$$

[7] Туральф Альберт Ско́лем (Скулем) (Thoralf Albert Skolem; 1887—1963) — норвежский математик, логик и философ.

[8] Жак Эрбра́н (Jacques Herbrand; 1908—1931) — французский математик.

[9] от *лат.* resolutio — решение.

[10] Джон Алан Ро́бинсон (John Alan Robinson; род. в 1930 г.) — американский математик и информатик.

где $\mathfrak{D}_1, \ldots, \mathfrak{D}_p$ — дизъюнкции конечного числа пропозициональных переменных или их отрицаний. Тем самым мы формируем множество дизъюнктов $\mathscr{K} = \{\mathfrak{D}_1, \ldots, \mathfrak{D}_p\}$.

Два дизъюнкта этого множества $\mathfrak{D}_i$ и $\mathfrak{D}_j$, содержащие пропозициональные переменные с противоположными знаками, — *контрарные*[11] *литералы* (к примеру Y и $\overline{Y}$), и, следовательно, $\mathfrak{D}_i = \mathfrak{D}'_i \vee Y$, $\mathfrak{D}_j = \mathfrak{D}'_j \vee \overline{Y}$ формируют третий дизъюнкт — *резольвенту*[12] $\mathfrak{D}'_i \vee \mathfrak{D}'_j$, в которой исключены контрарные литералы:

$$\frac{\mathfrak{D}'_i \vee Y,\ \mathfrak{D}'_j \vee \overline{Y}}{\mathfrak{D}'_i \vee \mathfrak{D}'_j}(\mathrm{R}).$$

В частности, если $\mathfrak{D}_i = Y$ и $\mathfrak{D}_j = \overline{Y}$, то резольвента для них — это дизъюнкт, равный 0. Его называют *пустым дизъюнктом* или *пустой резольвентой* и обозначают знаком «$\square$».

Резольвента $\mathfrak{D}'_i \vee \mathfrak{D}'_j$ — это логическое следствие дизъюнктов $\mathfrak{D}_i = \mathfrak{D}'_i \vee Y$ и $\mathfrak{D}_j = \mathfrak{D}'_j \vee \overline{Y}$, т. е.

$$\mathfrak{D}_i, \mathfrak{D}_j \vdash (\mathfrak{D}'_i \vee \mathfrak{D}'_j).$$

Неоднократно применяя правило получения резольвент *(принцип резолюции)* к множеству дизъюнктов, стремятся получить пустую резольвенту $\square$. Наличие $\square$ свидетельствует о получении противоречия, поскольку пустая резольвента получается из двух противоречащих друг другу литералов Y и $\overline{Y}$, каждый из которых является логическим следствием формулы $\overline{\mathfrak{A}}$ в соответствии с правилами $\dfrac{\mathfrak{A} \,\&\, \mathfrak{B}}{\mathfrak{A}}$ и $\dfrac{\mathfrak{A} \,\&\, \mathfrak{B}}{\mathfrak{B}}$.

Доказательство (от противного). если $\Gamma, \overline{\mathfrak{A}} \vdash 0$, где Γ — множество формул, то $\Gamma \vdash \mathfrak{A}$. Следовательно, получение противоречия с помощью формулы $\overline{\mathfrak{A}}$ означает общезначимость формулы $\mathfrak{A}$, что и требовалось доказать.

Изложим по шагам алгоритм метода резолюций для логики высказываний:

Шаг 1. Принять отрицание формулы $\mathfrak{A}$, т. е. $\overline{\mathfrak{A}}$.

[11] от *лат.* contrarius — противоположный.

[12] от *лат.* resolvo — решаю.

Шаг 2. Привести формулу $\overline{\mathfrak{A}}$ к КНФ.

Шаг 3. Выписать множество её дизъюнктов $\mathscr{K} = \{\mathfrak{D}_1, \ldots, \mathfrak{D}_p\}$.

Шаг 4. Выполнить анализ пар множества $\mathscr{K}$ по правилу: если существуют дизъюнкты $\mathfrak{D}_i$ и $\mathfrak{D}_j$, один из которых ($\mathfrak{D}_i$) содержит некоторый литерал X, а другой ($\mathfrak{D}_j$) — контрарный литерал $\overline{X}$, то нужно соединить эту пару с помощью дизъюнкции $(\mathfrak{D}_i \vee \mathfrak{D}_j)$ и сформировать новый дизъюнкт — резольвенту, исключив контрарные литералы X и $\overline{X}$.

Шаг 5. Если в результате соединения дизъюнктов, содержащих контрарные литералы, будет получена пустая резольвента $\square$, то результат достигнут (доказательство подтвердило противоречие), в противном случае включить резольвенту в множество дизъюнктов $\mathscr{K}$ и перейти к шагу 4.

При реализации указанного алгоритма возможны три случая:

1. Среди текущего множества дизъюнктов нет резольвируемых. Это означает, что формула $\mathfrak{A}$ не является общезначимой.

2. На каком-то шаге получается пустая резольвента. Формула $\mathfrak{A}$ общезначима.

3. Процесс не останавливается, т. е. множество дизъюнктов пополняется всё новыми резольвентами, среди которых нет пустых. В таком случае нельзя ничего сказать об общезначимости формулы $\mathfrak{A}$.

Метод резолюций годен и для доказательства того, что формула $\mathfrak{B}$ является логическим следствием формул $\mathfrak{A}_1, \ldots, \mathfrak{A}_n$, поскольку

$$\mathfrak{A}_1, \ldots, \mathfrak{A}_n \vdash \mathfrak{B} \;\equiv\; \vdash \mathfrak{A}_1 \,\&\, \ldots \,\&\, \mathfrak{A}_n \to \mathfrak{B}.$$

Для того чтобы применить метод резолюций для данного случая, нужно воспользоваться следующим равенством:

$$\mathfrak{A}_1 \,\&\, \ldots \,\&\, \mathfrak{A}_n \to \mathfrak{B} \;\equiv\; \overline{\mathfrak{A}_1 \,\&\, \ldots \,\&\, \mathfrak{A}_n \,\&\, \overline{\mathfrak{B}}}.$$

Формула $\mathfrak{C} = \overline{\mathfrak{A}_1 \,\&\, \ldots \,\&\, \mathfrak{A}_n \,\&\, \overline{\mathfrak{B}}}$ будет общезначимой, если формула $\overline{\mathfrak{C}} = \mathfrak{A}_1 \,\&\, \ldots \,\&\, \mathfrak{A}_n \,\&\, \overline{\mathfrak{B}}$ является противоречием. Далее применяем описанный алгоритм метода резолюций.

2.5.2. Метод резолюций в логике предикатов

Предположим, что дана формула логики предикатов $\mathfrak{A}$. Как проверить общезначимость этой формулы? Применим ли метод резолюций? Применим, но с некоторыми дополнениями.

Если формула $\mathfrak{A}$ не содержит предикатов и знаков функций, то мы фактически имеем формулу логики высказываний, поскольку предикаты в данном случае можно рассматривать как пропозициональные переменные. Следовательно, метод, резолюций не требует существенных изменений.

В случае, если формула $\mathfrak{A}$ содержит кванторы, то её надо привести к ПНФ, а затем устранить кванторы существования, вводя сколемовские функции. При этом, естественно, появятся знаки функций. Затем полученную формулу приводят к конъюнктивной нормальной форме и ищут резольвенты для дизъюнктов.

Что делать, если два контрарных литерала, в данном случае они имеют вид $P(\dots)$ и $\overline{P(\dots)}$, содержат два разных терма? Например, $\mathfrak{D}_i = \mathfrak{D}'_i \vee P(a,x)$ и $\mathfrak{D}_j = \mathfrak{D}'_j \vee \overline{P\big(a, f(b)\big)}$. Под *термом*[13] здесь и далее будем понимать переменную, константу или функциональный символ — всё то, что может выражать объекты из предметной области. В этом случае необходимо провести их *унификацию*[14] $\big\langle x \mid f(b)\big\rangle$, т. е. замену термов. Символическая запись $\mathfrak{A}(t_1)\langle t_1 \mid t_2\rangle$ означает замену в формуле $\mathfrak{A}$ терма t_1 термом t_2. В нашем случае

$$\mathfrak{D}_i\big\langle x \mid f(b)\big\rangle = \big(\mathfrak{D}'_i \vee P(a,x)\big)\big\langle x \mid f(b)\big\rangle = \mathfrak{D}'_i\big\langle x \mid f(b)\big\rangle \vee P\big(a, f(b)\big),$$

$$\mathfrak{D}_j\big\langle x \,\big|\, f(b)\big\rangle = \Big(\mathfrak{D}'_j \vee \overline{P\big(a, f(b)\big)}\Big)\big\langle x \,\big|\, f(b)\big\rangle = \mathfrak{D}'_j\big\langle x \,\big|\, f(b)\big\rangle \vee \overline{P\big(a, f(b)\big)}.$$

Далее ищется резольвента $\dfrac{\mathfrak{D}_i,\ \mathfrak{D}_j}{\mathfrak{D}'_i\big\langle x \mid f(b)\big\rangle \vee \mathfrak{D}'_j\big\langle x \mid f(b)\big\rangle}$(R).

Сформулируем более строго правила замены термов и понятие унификации.

Замена — это пара вида $\langle x \mid t\rangle$, где x — переменная, а t — терм. Применение замены $\langle x \mid t\rangle$ к терму t_0, входящему в некоторую формулу, определяется индуктивно:

1) $x\langle x \mid t\rangle = t$, если x — переменная;

[13] от *лат.* terminus — выражение, слово, термин.

[14] от *лат.* unio — единственность, facere — делать.

2) $a\langle x \mid t\rangle = a$, если a — константа (фиксированный объект);

3) $f(t_1, \ldots, t_n)\langle x \mid t\rangle = f\big(t_1\langle x \mid t\rangle, \ldots, t_n\langle x \mid t\rangle\big)$.

Таким образом, *унификация двух последовательностей термов* $t_1, \ldots, t_n$ и $\tau_1, \ldots, \tau_n$ — это замена $\langle x \mid t\rangle$ такая, что

$$t_i\langle x \mid t\rangle = \tau_i\langle x \mid t\rangle \text{ при } i = 1, \ldots, n.$$

Резольвента для дизъюнктов $\mathfrak{D}_i = \mathfrak{D}'_i \vee P(t_1)$ и $\mathfrak{D}_j = \mathfrak{D}'_j \vee \overline{P(t_2)}$ ищется с помощью унификации термов t_1 и t_2:

$$\frac{\mathfrak{D}_i,\ \mathfrak{D}_j}{\mathfrak{D}'_i\langle x \mid t\rangle \vee \mathfrak{D}'_j\langle x \mid t\rangle}(\mathrm{R}).$$

2.6. Примеры решения задач

Пример 2.1. Записать символически высказывание:
Некоторые устройства укомплектованы амперметром.

Р е ш е н и е. Введём следующие обозначения:

$D(x)$ — предикат «x — устройство»;

$A(x)$ — предикат «x укомплектован амперметром».

Исходное высказывание запишется в следующем виде:

$$\exists x\big(D(x)\ \&\ A(x)\big).$$

О т в е т: $\exists x\big(D(x)\ \&\ A(x)\big)$.

Пример 2.2. Записать с использованием предиката равенства и функции умножения высказывание:
Если один из двух сомножителей делится на z, то на него делится и произведение.

Р е ш е н и е. Введём следующие обозначения:

$E(x, y)$ — предикат «x равен y»;

$f(x, y)$ — функция «произведение x и y».

С помощью сделанных обозначений опишем предикат $D(x, y)$ — «x делится на y без остатка»:

$$D(x, y) = \exists v\, E\big(x, f(y, v)\big).$$

С помощью предиката делимости можем записать исходное утверждение:

$$\forall x \forall y \forall z \Big(D(x, z)\ \&\ D(y, z) \to D\big(f(x, y), z\big) \Big).$$

Поскольку по условию задачи имеем право пользоваться только предикатом равенства и функцией умножения, заменим в полученной формуле предикат делимости:

$$\forall x \forall y \forall z \Big(\exists v\, E\big(x, f(z, v)\big) \vee \\ \vee\, \exists v\, E\big(y, f(z, v)\big) \to \exists v\, E\big(f(x, y), f(z, v)\big) \Big).$$

О т в е т:

$$\forall x \forall y \forall z \Big(\exists v\, E\big(x, f(z, v)\big) \vee \\ \vee\, \exists v\, E\big(y, f(z, v)\big) \to \exists v\, E\big(f(x, y), f(z, v)\big) \Big).$$

Пример 2.3. Найти сколемовскую нормальную форму следующей формулы:

$$\forall x \forall y \forall z \Big(\exists v\, E\big(x, f(z, v)\big) \vee \\ \vee\, \exists v\, E\big(y, f(z, v)\big) \to \exists v\, E\big(f(x, y), f(z, v)\big) \Big).$$

Р е ш е н и е. Выполним преобразования формулы:

$$\forall x \forall y \forall z \Big(\exists v\, E\big(x, f(z, v)\big) \vee \\ \vee\, \exists v\, E\big(y, f(z, v)\big) \to \exists v\, E\big(f(x, y), f(z, v)\big) \Big) \equiv$$

$$\equiv \forall x \forall y \forall z \Big(\overline{\exists v\, E\big(x, f(z,v)\big) \vee \exists v\, E\big(y, f(z,v)\big)} \vee$$
$$\vee \exists v\, E\big(f(x,y), f(z,v)\big) \Big) \equiv$$

$$\equiv \forall x \forall y \forall z \Big(\forall v\, \overline{E\big(x, f(z,v)\big)} \,\&\, \forall v\, \overline{E\big(y, f(z,v)\big)} \vee$$
$$\vee \exists v\, E\big(f(x,y), f(z,v)\big) \Big) \equiv$$

$$\equiv \forall x \forall y \forall z \Big(\exists v\, E\big(f(x,y), f(z,v)\big) \vee$$
$$\vee \forall u\, \overline{E\big(x, f(z,u)\big)} \,\&\, \forall u\, \overline{E\big(y, f(z,u)\big)} \Big) \equiv$$

$$\equiv \forall x \forall y \forall z \Big(\exists v\, E\big(f(x,y), f(z,v)\big) \vee$$
$$\vee \forall s\, \overline{E\big(x, f(z,s)\big)} \,\&\, \forall t\, \overline{E\big(y, f(z,t)\big)} \Big) \equiv$$

$$\equiv \forall x \forall y \forall z \exists v \forall s \forall t \Big(E\big(f(x,y), f(z,v)\big) \vee$$
$$\vee \overline{E\big(x, f(z,s)\big)} \,\&\, \overline{E\big(y, f(z,t)\big)} \Big).$$

Получили предварённую нормальную форму. Для получения сколемовской нормальной формы выполним замену $\langle v \mid g(x,y,z) \rangle$:

$$\forall x \forall y \forall z \forall s \forall t \Big(E\Big(f(x,y), f\big(z, g(x,y,z)\big)\Big) \vee$$
$$\vee \overline{E\big(x, f(z,s)\big)} \,\&\, \overline{E\big(y, f(z,t)\big)} \Big).$$

О т в е т:

$$\forall x \forall y \forall z \forall s \forall t \Big(E\Big(f(x,y), f\big(z, g(x,y,z)\big)\Big) \vee$$
$$\vee \overline{E\big(x, f(z,s)\big)} \,\&\, \overline{E\big(y, f(z,t)\big)} \Big).$$

Пример 2.4. Доказать с помощью метода резолюций, что формула $\overline{B}$ является логическим следствием формул $\overline{A \,\&\, B}$ и A.

Доказательство. Чтобы формула $\overline{B}$ была логическим следствием формул $\overline{A \,\&\, B}$ и A, должно иметь место равенство

$$\overline{A \,\&\, B} \,\&\, A \to \overline{B} \equiv 1.$$

Поскольку метод резолюций устанавливает тождественную ложность формул, задачу необходимо переформулировать в виде противоречия. Эквивалентными преобразованиями равенство можно привести к виду $\overline{A \,\&\, B} \,\&\, A \,\&\, B \equiv 0$, пригодному для применения метода резолюций. КНФ полученной формулы будет иметь вид:

$$(\overline{A} \vee \overline{B}) \,\&\, A \,\&\, B.$$

Выпишем набор дизъюнктов:

$$\mathscr{K} = \{\mathfrak{D}_1 = \overline{A} \vee \overline{B}; \quad \mathfrak{D}_2 = A; \quad \mathfrak{D}_3 = B\}.$$

Получим новые дизъюнкты:

$\mathfrak{D}_4 = \overline{B}$ — получен с помощью принципа резолюции для $\mathfrak{D}_1$ и $\mathfrak{D}_2$;

$\mathfrak{D}_5 = \square$ — получен с помощью принципа резолюции для $\mathfrak{D}_3$ и $\mathfrak{D}_4$.

В итоге мы получили пустой дизъюнкт. Следовательно, равенство $\overline{A \,\&\, B} \,\&\, A \,\&\, B \equiv 0$ является верным, а значит, верным также является равенство $\overline{A \,\&\, B} \,\&\, A \to \overline{B} \equiv 1$. Это, в свою очередь, означает, что исходное утверждение «формула $\overline{B}$ является логическим следствием формул $\overline{A \,\&\, B}$ и A» верно.

Пример 2.5. Доказать верность умозаключения:
Две рощи никогда не похожи одна на другую. Сосны и ели выглядят совершенно одинаково. Значит, сосны и ели не являются двумя рощами.

Доказательство. Введём следующие обозначения:

$G(x)$ — предикат «x — роща»;

$P(x)$ — предикат «x — сосна»;

$F(x)$ — предикат «x — ель»;

$L(x,y)$ — предикат «x похож на y».

Условие задачи можно теперь записать так:

$\forall x\forall y\big(G(x)\,\&\,G(y)\to\overline{L(x,y)}\big)$ — «Две рощи никогда не похожи одна на другую»;

$\forall x\forall y\big(P(x)\,\&\,F(y)\to L(x,y)\big)$ — «Сосны и ели выглядят совершенно одинаково»;

$\forall x\forall y\big(P(x)\,\&\,F(y)\to\overline{G(x)\,\&\,G(y)}\big)$ — «Сосны и ели не являются двумя рощами».

Последнее утверждение можно также записать как

$$\overline{\exists x\exists y\big(P(x)\,\&\,F(y)\,\&\,G(x)\,\&\,G(y)\big)}$$

или выполнить эквивалентные преобразования.

Необходимо доказать, что

$$\Big(\forall x\forall y\big(G(x)\,\&\,G(y)\to\overline{L(x,y)}\big)\&\\ \&\,\forall x\forall y\big(P(x)\,\&\,F(y)\to L(x,y)\big)\Big)\to\\ \to\overline{\exists x\exists y\big(P(x)\,\&\,F(y)\,\&\,G(x)\,\&\,G(y)\big)}\equiv 1.$$

Преобразуем равенство к виду противоречия и приведём к сколемовской нормальной форме:

$$\forall x\forall y\big(G(x)\,\&\,G(y)\to\overline{L(x,y)}\big)\&\\ \&\,\forall x\forall y\big(P(x)\,\&\,F(y)\to L(x,y)\big)\&\\ \&\,\exists x\exists y\big(P(x)\,\&\,F(y)\,\&\,G(x)\,\&\,G(y)\big)\equiv 0;$$

$$\forall x\forall y\big(\overline{G(x)}\vee\overline{G(y)}\vee\overline{L(x,y)}\big)\&\\ \&\,\forall x\forall y\big(\overline{P(x)}\vee\overline{F(y)}\vee L(x,y)\big)\&\\ \&\,\exists u\exists v\big(P(u)\,\&\,F(v)\,\&\,G(u)\,\&\,G(v)\big)\equiv 0;$$

$$\exists u\exists v\forall x\forall y\forall x\forall y\Big(\big(P(u)\,\&\,F(v)\,\&\,G(u)\,\&\,G(v)\big)\&\\ \&\,\big(\overline{G(x)}\vee\overline{G(y)}\vee\overline{L(x,y)}\big)\,\&\,\big(\overline{P(x)}\vee\overline{F(y)}\vee L(x,y)\big)\Big)\equiv 0.$$

После замен $\langle u \mid a\rangle$, $\langle v \mid b\rangle$ получим сколемовскую нормальную форму:

$$\forall x\forall y\forall x\forall y\Big(\big(P(a)\,\&\,F(b)\,\&\,G(a)\,\&\,G(b)\big)\&\\ \&\,\big(\overline{G(x)}\vee\overline{G(y)}\vee\overline{L(x,y)}\big)\,\&\,\big(\overline{P(x)}\vee\overline{F(y)}\vee L(x,y)\big)\Big)\equiv 0.$$

Имеем набор дизъюнктов:

$$\mathscr{K}=\big\{\mathfrak{D}_1=\overline{G(x)}\vee\overline{G(y)}\vee\overline{L(x,y)};\quad \mathfrak{D}_2=\overline{P(x)}\vee\overline{F(y)}\vee L(x,y);\\ \mathfrak{D}_3=P(a);\quad \mathfrak{D}_4=F(b);\quad \mathfrak{D}_5=G(a);\quad \mathfrak{D}_6=G(b)\big\}.$$

Получим новые дизъюнкты:

$\mathfrak{D}_7=\overline{G(y)}\vee\overline{L(a,y)}$ — получен с помощью принципа резолюции для $\mathfrak{D}_1$ и $\mathfrak{D}_5$ с заменой $\langle x \mid a\rangle$;

$\mathfrak{D}_8=\overline{L(a,b)}$ — получен с помощью принципа резолюции для $\mathfrak{D}_6$ и $\mathfrak{D}_7$ с заменой $\langle y \mid b\rangle$;

$\mathfrak{D}_9=\overline{F(y)}\vee L(a,y)$ — получен с помощью принципа резолюции для $\mathfrak{D}_2$ и $\mathfrak{D}_3$ с заменой $\langle x \mid a\rangle$;

$\mathfrak{D}_{10}=L(a,b)$ — получен с помощью принципа резолюции для $\mathfrak{D}_4$ и $\mathfrak{D}_9$ с заменой $\langle y \mid b\rangle$;

Наконец, $\mathfrak{D}_{11}=\square$ — получен с помощью принципа резолюции для $\mathfrak{D}_8$ и $\mathfrak{D}_{10}$.

Метод резолюций сошёлся на пустом дизъюнкте. Следовательно, исходное утверждение верно.

Пример 2.6. Доказать верность умозаключения $\dfrac{\exists x\forall y\,P(x,y)}{\forall y\exists x\,P(x,y)}$.

Доказательство. Докажем, что

$$\exists x\forall y\,P(x,y)\to\forall y\exists x\,P(x,y)\equiv 1.$$

Для этого сначала выполним преобразования:

$$\overline{\exists x \forall y\, P(x,y)} \vee \forall y \exists x\, P(x,y) \equiv 1;$$
$$\overline{\overline{\exists x \forall y\, P(x,y)} \vee \forall y \exists x\, P(x,y)} \equiv 0;$$
$$\exists x \forall y\, P(x,y)\ \&\ \overline{\forall y \exists x\, P(x,y)} \equiv 0;$$
$$\exists x \forall y\, P(x,y)\ \&\ \exists y \forall x\, \overline{P(x,y)} \equiv 0;$$
$$\exists x \forall y\, P(x,y)\ \&\ \exists z \forall u\, \overline{P(u,z)} \equiv 0;$$
$$\exists x \forall y \exists z \forall u \big(P(x,y)\ \&\ \overline{P(u,z)}\big) \equiv 0.$$

Получили ПНФ. После замен $\langle x \mid a \rangle$ и $\big\langle z \;\big|\; f(y) \big\rangle$ получим сколемовскую нормальную форму:

$$\forall y \forall u \Big(P(a,y)\ \&\ \overline{P\big(u, f(y)\big)}\Big) \equiv 0.$$

Воспользуемся методом резолюций. Имеем набор дизъюнктов:

$$\mathscr{K} = \Big\{\mathfrak{D}_1 = P(a,y); \quad \mathfrak{D}_2 = \overline{P\big(u, f(y)\big)}\Big\}.$$

Получим пустой дизъюнкт $\mathfrak{D}_3 = \square$ с помощью принципа резолюции для $\mathfrak{D}_1$ и $\mathfrak{D}_2$ с заменами $\big\langle y \;\big|\; f(y) \big\rangle$ и $\langle u \mid a \rangle$.

Метод сошёлся на пустом дизъюнкте. Следовательно, исходное умозаключение верно.

Глава 3

Формальные теории

3.1. Определение формальной теории

Формальная теория (исчисление) — это способ изложения логики без приписывания пропозициональным переменным, предикатам и формулам какого-либо значения. По замыслу создателей формальных теорий таким образом можно избежать многих неприятностей, возникающих при использовании в логике человеческого языка, допускающего двусмысленность, недосказанность, переиначивание исходно заложенного значения, смысла и т. д.

Формальная теория — это игра со знаками, со словами и предложениями, составленными из знаков. При этом имеется в виду, что правила составления слов из знаков и правила игры со словами и предложениями заранее оговорены, точно и строго прописаны. В основе формальной теории — язык, на котором «разговаривает» теория. На первое место выходит синтаксис этого языка, т. е. способ построения формул, в противопоставление семантике.

Имеет место следующее определение формальной теории. Формальная теория $\mathscr{T}$ состоит из следующих компонент:

1. Множества знаков, образующих *алфавит* языка теории.

2. Множества слов, составленных из знаков алфавита, называемых *формулами.*

3. Подмножества формул всего множества формул, называемых *аксиомами*[1)].

4. Множества *правил вывода,* с помощью которых из формул получают новые формулы.

В язык теории $\mathscr{T}$ входит алфавит и формулы. Количество аксиом может быть конечным или бесконечным. В последнем случае для наглядного представления они задаются с помощью *схем*[2)].

[1)] от *греч.* ἀξίωμα — требование, положение.

[2)] от *греч.* σχήμα — образ, вид.

По схемам легко выписываются сами аксиомы. Под логическими аксиомами, как правило, понимают аксиомы *базовой логики,* над которой надстраиваются конкретные теории за счёт добавления новых аксиом, отражающих специфику конкретной теории. Такие аксиомы называют *нелогическими.*

Обычно формулы состоят из конечного числа знаков. Но бывают теории с бесконечно длинными формулами, т. е. с формулам, содержащими бесконечное число знаков.

Для обозначения формул будем пользоваться готическими буквами $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}$ и т. д. Эти буквы не входят в алфавит теории, они представляют собой только условные обозначения формул.

Формальное доказательство формулы $\mathfrak{A}$ в теории $\mathscr{T}$ — это конечная последовательность формул $\mathfrak{A}_1, \mathfrak{A}_2, \ldots, \mathfrak{A}_n$, где каждая формула $\mathfrak{A}_i$ — либо аксиома, либо получена из предыдущих с помощью одного из правил вывода, а $\mathfrak{A}_n$ — это формула $\mathfrak{A}$. Формула $\mathfrak{A}$ в этом случае называется *теоремой*[3], для теорем используется запись $\vdash \mathfrak{A}$.

Формальная теория называется *полной,* если для всякой формулы $\mathfrak{A}$ имеем $\vdash \mathfrak{A}$ или $\vdash \overline{\mathfrak{A}}$.

По замыслу создателя исчисления предикатов Г. Фреге[4], любая правильно построенная формула, а точнее высказывание (высказывание в логике предикатов — это *закрытая формула,* т. е. формула, все переменные которой связаны кванторами), должна быть теоремой, т. е. должна быть доказываемым утверждением. Иначе говоря, исчисление высказываний и исчисление предикатов должны быть полными теориями. Ожидания Фреге оправдались. Но, как оказалось, более сильные теории, включающие арифметику, неполны. Это было доказано К. Гёделем[5] (см. стр. 48).

Формальная теория называется *непротиворечивой,* если в ней не является доказуемой формула $A \,\&\, \overline{A}$, где A — произвольное высказывание теории.

Смысл условия непротиворечивости в том, что оно может быть доказано средствами и в рамках самой формальной теории. Увы, таким способом, как выяснилось, невозможно установить даже непротиворечивость (формальной) арифметики.

---

[3] от *греч.* θεώρημα — представление, положение.

[4] Фридрих Людвиг Готлоб Фре́ге (Friedrich Ludwig Gottlob Frege; 1848—1925) — немецкий логик, математик и философ.

[5] Курт Фридрих Гёдель (Kurt Friedrich Gödel; 1906—1978) — австрийский логик, математик и философ.

3.2. Исчисление высказываний

Исчисление высказываний — это формальная теория, определяемая следующим образом:

1. Алфавит:
   1) пропозициональные переменные $A, B, \ldots, X_1, X_2, \ldots$;
   2) пропозициональные связки «&», «$\vee$», «$\overline{\ }$», «$\to$»;
   3) скобки «(», «)».
2. Определение формулы исчисления высказываний:
   1) пропозициональная переменная есть формула;
   2) если $\mathfrak{A}$ и $\mathfrak{B}$ — формулы, то $(\mathfrak{A} \,\&\, \mathfrak{B})$, $(\mathfrak{A} \vee \mathfrak{B})$, $(\mathfrak{A} \to \mathfrak{B})$ — формулы;
   3) если $\mathfrak{A}$ — формула, то $\overline{\mathfrak{A}}$ — формула.
3. Аксиомы[6]:

I. 1) $A \to (B \to A)$;
   2) $\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)$.

II. 1) $(A \,\&\, B) \to A$;
   2) $(A \,\&\, B) \to B$;
   3) $(A \to B) \to \Big((A \to C) \to \big(A \to (B \,\&\, C)\big)\Big)$.

III. 1) $A \to (A \vee B)$;
   2) $B \to (A \vee B)$;
   3) $(A \to C) \to \Big((B \to C) \to \big((A \vee B) \to C\big)\Big)$.

IV. 1) $(A \to B) \to (\overline{B} \to \overline{A})$;
   2) $A \to \overline{\overline{A}}$;
   3) $\overline{\overline{A}} \to A$.

[6] Возможны и другие системы аксиом. Приведённые здесь аксиомы взяты из [14, с. 72].

4. Правила вывода:

   1) *Правило подстановки.*
   Пусть $\mathfrak{A}$ — формула, содержащая некоторую переменную A. Тогда, если $\mathfrak{A}$ — выводимая формула, то, заменив в ней переменную A всюду, где она входит, произвольной формулой $\mathfrak{B}$, мы также получим выводимую формулу.
   2) *Правило заключения* (modus ponens).
   Если $\mathfrak{A}$ и $(\mathfrak{A} \to \mathfrak{B})$ — выводимые формулы, то $\mathfrak{B}$ — также выводимая формула:

$$\frac{\mathfrak{A},\ \mathfrak{A} \to \mathfrak{B}}{\mathfrak{B}}.$$

Рассмотрим определение доказательства в исчислении высказываний.

Пусть Γ — множество формул. Формула $\mathfrak{A}$ *выводима* из множества *гипотез*[7] Γ, если существует конечная последовательность формул $\mathfrak{A}_1, \mathfrak{A}_2, \ldots, \mathfrak{A}_n$, где каждая формула $\mathfrak{A}_i$ является либо аксиомой, либо гипотезой, либо получена из предыдущих с помощью правила вывода, и $\mathfrak{A}_n$ — это формула $\mathfrak{A}$.

Последовательность $\mathfrak{A}_1, \mathfrak{A}_2, \ldots, \mathfrak{A}_n$ — это *вывод* или *доказательство* формулы $\mathfrak{A}$ из множества гипотез Γ. Используется обозначение для вывода: $\Gamma \vdash \mathfrak{A}$.

Если $\Gamma = \varnothing$, то вывод называется доказательством формулы $\mathfrak{A}$, а сама формула $\mathfrak{A}$ в $\mathfrak{A}_1, \mathfrak{A}_2, \ldots, \mathfrak{A}_n$ называется *теоремой.* Используется символическая запись для теорем: $\vdash \mathfrak{A}$.

В исчислении высказываний имеет место следующая теорема.

Т е о р е м а 3 (дедукции). $\Gamma, \mathfrak{A} \vdash \mathfrak{B}$ *тогда и только тогда, когда* $\Gamma \vdash (\mathfrak{A} \to \mathfrak{B})$.

Отметим (без доказательства), что исчисление высказываний является полной и непротиворечивой теорией.

3.3. Исчисление предикатов

Исчисление предикатов — это формальная теория, получаемая за счёт добавления к исчислению высказываний новых знаков, понятия терма, новых типов формул и новых правил вывода.

[7] от *греч.* ὑπόθεσις — основание, предположение.

1. Новый алфавит:

   1) знаки индивидуальных переменных $x_1, x_2, \ldots, x_n, \ldots$;

   2) знаки индивидуальных констант $c_1, c_2, \ldots, c_n, \ldots$;

   3) знаки предикатов $P_1^{n_1}\underbrace{(\cdot, \ldots, \cdot)}_{n_1 \text{ мест}}$, $P_2^{n_2}\underbrace{(\cdot, \ldots, \cdot)}_{n_2 \text{ мест}}$, $\ldots$;

   4) знаки операций (функции)

   $$f_1^{n_1}\underbrace{(\cdot, \ldots, \cdot)}_{n_1 \text{ мест}}, \quad f_2^{n_2}\underbrace{(\cdot, \ldots, \cdot)}_{n_2 \text{ мест}}, \ \ldots;$$

   5) логические знаки (кванторы) «$\forall$», «$\exists$».

2. Термы:

   1) переменные $x_1, x_2, \ldots, x_n, \ldots$ — это термы;

   2) константы $c_1, c_2, \ldots, c_n, \ldots$ — это термы;

   3) если $f_m^n(\cdot, \ldots, \cdot)$ — n-местный знак операции, $t_1, \ldots, t_n$ — термы, то $f_m^n(t_1, \ldots, t_n)$ — терм.

3. Формулы:

   1) если $P_m^n(\cdot, \ldots, \cdot)$ — n-местный знак предиката, $t_1, \ldots, t_n$ — термы, то $P_m^n(t_1, \ldots, t_n)$ — формула;

   2) если $\mathfrak{A}$ и $\mathfrak{B}$ — формулы, то $(\mathfrak{A}\ \&\ \mathfrak{B})$, $(\mathfrak{A} \vee \mathfrak{B})$, $(\mathfrak{A} \to \mathfrak{B})$ — формулы;

   3) если $\mathfrak{A}$ — формула, то $\overline{\mathfrak{A}}$ — формула;

   4) если $\mathfrak{A}(x)$ — формула, содержащая переменную x, то $\forall x\, \mathfrak{A}(x)$, $\exists x\, \mathfrak{A}(x)$ — формулы.

4. Дополнительные аксиомы (аксиомы Бернайса[8)]):

   1′) $\forall x\, \mathfrak{A}(x) \to \mathfrak{A}(t)$;

   2′) $\mathfrak{A}(t) \to \exists x\, \mathfrak{A}(x)$.

   Здесь $t = t(x_1, \ldots, x_n)$ — терм, а в формуле $\mathfrak{A}$ нет кванторов, связывающих переменные $x_1, \ldots, x_n$.

[8)] Исаак Пауль Бернáйс (Isaak Paul Bernays; 1888—1977) — швейцарский математик.

5. Дополнительные правила вывода:

1) $\dfrac{\mathfrak{B} \to \mathfrak{A}(x)}{\mathfrak{B} \to \forall x\, \mathfrak{A}(x)}$ — правило обобщения;

2) $\dfrac{\mathfrak{A}(x) \to \mathfrak{B}}{\exists x\, \mathfrak{A}(x) \to \mathfrak{B}}$ — правило конкретизации.

Приведённый язык исчисления предикатов образует теорию, которую называют *узким исчислением предикатов* или *чистым исчислением предикатов.* Появление в теории аксиом, разрешающих «навешивание» кванторов по знакам операций или предикатов, приводит к *исчислениям высших порядков.* Исчисление предикатов — это *теория первого порядка.*

Расширение узкого исчисления предикатов за счёт добавления новых знаков предметных констант, знаков функций, знаков операций, знаков предикатов, новых аксиом, связывающих новые знаки, и новых правил вывода часто называют *прикладным исчислением предикатов.*

Также без доказательства отметим, что исчисление предикатов является полной и непротиворечивой теорией.

3.4. Формальная арифметика

Теория, содержащая исчисление предикатов, называется *эгалитарной*[9], если она имеет дополнительный двухместный предикат равенства $=(\cdot,\cdot)$, для которого выполняются две *нелогические аксиомы равенства* (здесь $\mathfrak{A}$ — произвольная формула):

$1^*)\ \ \forall x\big(=(x,x)\big);$

$2^*)\ \ =(x,y) \to \big(\mathfrak{A}(\ldots,x,\ldots,y,\ldots) \to \mathfrak{A}(\ldots,y,\ldots,x,\ldots)\big).$

Для обозначения предиката равенства вместо префиксной формы записи $=(x,y)$ обычно используют инфиксную форму: $x = y$. Таким образом, эгалитарная теория — это просто теория первого порядка с равенством.

Формальная арифметика — это эгалитарная теория, в которой имеются следующие компоненты:

[9] от *фр.* égalité — равенство.

1. Предметная константа «$\emptyset$» (ноль).

2. Двухместные операции сложения «$+$» и умножения «$\times$», а также одноместная операция инкремента[10] «$'$».

3. Знак равенства «$=$».

4. Нелогические аксиомы равенства (см. выше) и следующие нелогические аксиомы арифметики:

$$
\begin{aligned}
&3^*)\ \Big(\mathfrak{A}(\emptyset)\ \&\ \forall x\big(\mathfrak{A}(x)\to\mathfrak{A}(x')\big)\Big)\to\forall x\,\mathfrak{A}(x),\\
&4^*)\ (t_1'=t_2')\to(t_1=t_2),\\
&5^*)\ (t_1=t_2)\to(t_1'=t_2'),\\
&6^*)\ (t_1=t_2)\to\big((t_2=t_3)\to(t_1=t_3)\big),\\
&7^*)\ \overline{t'=\emptyset},\\
&8^*)\ (t+\emptyset)=t,\\
&9^*)\ (t\times\emptyset)=\emptyset,\\
&10^*)\ (t_1+t_2')=(t_1+t_2)',\\
&11^*)\ (t_1\times t_2')\to\big((t_1\times t_2)+t_1\big),
\end{aligned}
$$

где $\mathfrak{A}$ — любая формула, а t, t_1 и t_2 — любые термы.

Аксиома № 3 — это известный способ доказательства посредством *математической индукции*[11]. Если вместо t' написать $t+1$, то ясно, что t' — это следующее натуральное число, идущее за t. Другими словами, аксиомы арифметики определяют натуральные числа и правила оперирования с ними с помощью операций сложения и умножения.

Метод математической индукции, который входит в качестве аксиомы в формальную арифметику, может быть усилен за счёт расширения области его применения до так называемых *трансфинитных*[12] *чисел,* которые, как видно из их названия, «идут следом» за финитными, т. е. натуральными числами. Получается более мощный способ доказательства теорем, названный *методом трансфинитной индукции.*

[10] от *лат.* incrementum — рост, увеличение.

[11] от *лат.* inductio — наведение, побуждение.

[12] от *лат.* trans — сквозь, через, за и finitus — ограниченный.

Непротиворечивость формальной арифметики доказывается в более широкой формальной теории, содержащей арифметику и принцип трансфинитной индукции.

Т е о р е м а 4 (Гёделя о неполноте). *Во всякой теории первого порядка, включающей формальную арифметику:*

1) *существует такая (истинная) формула* $\mathfrak{A}$*, что ни* $\mathfrak{A}$*, ни* $\overline{\mathfrak{A}}$ *не являются доказуемыми;*

2) *утверждение о непротиворечивости этой теории — это формула данной теории, которая не является доказуемой.*

Теорема Гёделя о неполноте дедуктивных систем говорит о том, что в любой достаточно богатой теории существуют высказывания, которые воспринимаются как истинные и разумные, но тем не менее они не могут быть доказаны теми средствами, которые предоставляет теория. По существу, теорема утверждает, что в любом мире есть вещи, познание которых требует выхода в более обширный, высший мир. Над каждой теорией нужно надстраивать более изощрённую метатеорию, над метатеорией — метаметатеорию и т. д. Собственно говоря, так вышло с доказательством непротиворечивости формальной арифметики (см. выше).

Кроме того, теорема Гёделя говорит о несостоятельности идеи полной формализации процесса логического вывода. Иначе говоря, не всё может быть формализовано, как бы этого ни хотелось математикам.

3.5. Примеры решения задач

Пример 3.1. Доказать выводимость формулы $A \to A$ в исчислении высказываний.

Д о к а з а т е л ь с т в о. Для доказательства выводимости необходимо выстроить такую последовательность формул, каждая из которых является либо аксиомой, либо получена из предыдущих применением правил вывода. При этом последней в этом наборе формул должна быть формула, выводимость которой доказываем. Имеем следующую последовательность формул:

$\mathfrak{A}_1 = A \to \big((A \to A) \to A\big)$ — получена из аксиомы I.1 с помощью подстановки $B := (A \to A)$.

$\mathfrak{A}_2 = \Big(A \to \big((A \to A) \to A\big)\Big) \to \Big(\big(A \to (A \to A)\big) \to (A \to A)\Big)$ — получена из аксиомы I.2 с помощью подстановок $B := (A \to A)$ и $C := A$.

$\mathfrak{A}_3 = \big(A \to (A \to A)\big) \to (A \to A)$ — получена с помощью правила заключения для формул $\mathfrak{A}_1$ и $\mathfrak{A}_2$.

$\mathfrak{A}_4 = A \to (A \to A)$ — получена из аксиомы I.1 с помощью подстановки $B := A$.

$\mathfrak{A}_5 = A \to A$ — получена с помощью правила заключения для формул $\mathfrak{A}_4$ и $\mathfrak{A}_3$.

Выводимость доказана.

Пример 3.2. Доказать, что в исчислении высказываний из гипотез $A \to B$ и $B \to C$ выводится формула $A \to C$.

Доказательство. Для доказательства воспользуемся теоремой о дедукции, т. е. вместо $A \to B,\ B \to C \vdash A \to C$ докажем $A \to B,\ B \to C,\ A \vdash C$:

$\mathfrak{A}_1 = A \to B$ — первая гипотеза.

$\mathfrak{A}_2 = B \to C$ — вторая гипотеза.

$\mathfrak{A}_3 = A$ — третья гипотеза.

$\mathfrak{A}_4 = B$ — получена с помощью правила заключения для формул $\mathfrak{A}_1$ и $\mathfrak{A}_3$.

$\mathfrak{A}_5 = C$ — получена с помощью правила заключения для формул $\mathfrak{A}_2$ и $\mathfrak{A}_4$.

Выводимость доказана.

Пример 3.3. Доказать, что формула $\forall x\, P(x) \to \exists x\, P(x)$ является теоремой исчисления предикатов.

Доказательство. Для доказательства построим вывод этой формулы из аксиом:

$\mathfrak{A}_1 = \forall x\, P(x) \to P(y)$ — получена из аксиомы Бернайса $1'$ с помощью подстановок $\mathfrak{A}(x) := P(x)$ и $t := y$.

$\mathfrak{A}_2 = P(y) \to \exists x\, P(x)$ — получена из аксиомы Бернайса $2'$ с помощью подстановок $\mathfrak{A}(x) := P(x)$ и $t := y$.

$\mathfrak{A}_3 = \forall x\, P(x) \to \exists x\, P(x)$ — получена из правила

$$A \to B,\ B \to C \vdash A \to C$$

(см. предыдущий пример) с помощью подстановок $A := \forall x\, P(x)$, $B := P(y)$ и $C := \exists x\, P(x)$, что и требовалось доказать.

Глава 4

Теория алгоритмов

4.1. Алгоритмы и вычислимые функции

Алгоритм[1)] — это раз и навсегда составленная программа, в которой всё заранее предусмотрено. Выражаясь популярно, алгоритм — это точное, воспроизводимое, поддающееся исполнению предписание, определяющее — шаг за шагом — каким путём надлежит решать данную задачу. Алгоритмом является любое формализованное доказательство математической теоремы, равно как и программа цифровой машины, переводящей с одного языка на другой.

Алгоритмом принято называть систему вычислений, которая для некоторого класса математических задач из записи A «условий» задачи позволяет при помощи однозначно определённой последовательности операций, совершаемых «механически», без вмешательства творческих способностей человека, получить запись B «решений» задачи.

Таким образом, алгоритм — это процедура Ψ, которая:

1. Применяется к строго определённым исходным данным A. Её нельзя применять к другому типу данных, она либо будет не способна с ними что-то сделать, либо может выдать бессмысленный результат, т. е. результат, не поддающийся анализу с точки зрения тех ожиданий, которые связывались с алгоритмом при его создании.

2. Разделена на отдельные простые шаги; каждый шаг состоит в непосредственной обработке возникшего к этому шагу состояния S в состояние $S^* = \Omega_\Psi(S)$.

3. Непосредственная переработка S в $S^* = \Omega_\Psi(S)$ производится однозначно заданным способом лишь на основании информации о виде заранее ограниченной «активной» части состояния S и затрагивает лишь эту активную часть.

[1)] алгорифм *(устар.);* от имени Мухаммада ибн Мусы Хорезми́ (аль-Хорезми) (محمد بن موسى خوارزمى; ок. 780—ок. 850) — великого среднеазиатского математика, астронома и географа.

4. Процесс переработки $A_0 = A$ в $A_1 = \Omega_\Psi(A_0)$, A_1 в $A_2 = \Omega_\Psi(A_1)$ и т. д. продолжается до тех пор, пока либо не произойдет безрезультатная остановка (или оператор Ω_Ψ не определён для возникшего состояния), либо не появится сигнал о получении «решения». При этом не исключается возможность непрекращающегося процесса переработки.

Каждый алгоритм задаёт функцию, поскольку по набору исходных данных выдаётся результат применения алгоритма к этим данным. Естественно назвать функцию, значения которой могут находиться с помощью некоторого алгоритма, *вычислимой функцией*. Таким образом, вычислимая функция — это такая функция, для которой существует вычисляющий её значения алгоритм.

4.2. Машина Тьюринга

Первый важный и достаточно широкий класс алгоритмов был описан А. Тьюрингом[2)] и Э. Постом[3)] в 1936—1937 гг. Алгоритмы этого класса осуществляются абстрактными вычислительными машинами, называемыми в настоящее время *машинами Тьюринга—Поста* или просто *машинами Тьюринга*.

Машина Тьюринга — это абстрактный исполнитель алгоритма, состоящий из следующих компонент:

1. Бесконечная в обе стороны *лента*, разделённая на *ячейки*. Лента представляет собой *внешнюю память*. Каждая ячейка ленты находится в одном и только в одном *состоянии* из множества $\mathscr{A} = \{a_0, a_1, \ldots, a_m\}$, называемого *внешним алфавитом*. Состояние a_0 называется *пустым* и часто обозначается символом Λ.

2. *Внутренняя память*, принимающая одно из состояний, входящих в множество $\mathscr{Q} = \{q_0, q_1, \ldots, q_n\}$, называемое *внутренним алфавитом*. Состояние q_0 называется *«стоп»*.

[2)] Алан Матисон Тьюринг (Alan Mathison Turing; 1912—1954) — английский математик, логик и криптограф.

[3)] Эмиль Леон Пост (Emil Leon Post; 1897—1954) — американский математик и логик.

3. *Головка* чтения-записи, двигающаяся вдоль ленты и считывающая или записывающая содержимое ячейки, напротив которой она останавливается.

4. *Механического устройства,* передвигающего головку и меняющего состояния внешней и внутренней памяти. Если головка в состоянии q стоит напротив ячейки с номером k, то изменения состояния внутренней памяти и состояния ячейки происходят одновременно.

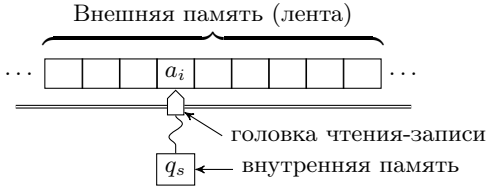

Рис. 4.1. Схема машины Тьюринга

Работа машины Тьюринга осуществляется посредством *команд,* которые выполняет механическое устройство. Команда имеет один из следующих трёх возможных видов:

1) $q_s a_i \to q_t a_j$,

2) $q_s a_i \to q_t a_j L$,

3) $q_s a_i \to q_t a_j R$,

где L — это движение головки влево на одну ячейку, а R — вправо. При этом всегда самый левый символ в записи команды $q_s \neq q_0$.

Смысл команд следующий. Если головка в состоянии q_s обозревает ячейку в состоянии a_i, то внутреннее состояние машины меняется на q_t, а ячейки — на a_j. Далее, в первом случае головка остаётся на месте, во втором — смещается на одну ячейку влево, в третьем — смещается на одну ячейку вправо.

Конечный набор команд образует *программу.*

Состояние машины Тьюринга — это последовательность состояний $a_{i_1}, \ldots, a_{i_r}$ ячеек ленты, состояние внутренней памяти q_s и номер k читаемой ячейки в состоянии a_{i_k}.

Состояние машины записывается в виде $a_{i_1} \dots a_{i_{k-1}} q_s a_{i_k} \dots a_{i_r}$ и называется *машинным словом* $\mathfrak{m}$ в алфавите $\mathscr{A} \cup \mathscr{Q}$. Символ q_s может быть самым левым, но не может быть самым правым в машинном слове, так как справа от него должно быть считываемое состояние ячейки.

Под воздействием программы происходит изменение состояния машины, сопровождающееся переделкой исходного машинного слова $\mathfrak{m} \to \mathfrak{m}^{(1)} \to \dots \to \mathfrak{m}^{(p)}$. В этом случае будем говорить, что машина $\mathfrak{T}$ перерабатывает слово $\mathfrak{m}$ в слово $\mathfrak{m}^{(p)}$, и обозначать этот факт $\mathfrak{m}^{(p)} = \mathfrak{T}(\mathfrak{m})$. Запись $\mathfrak{T}(\mathfrak{m})$ можно употреблять и в другом смысле — просто как обозначение машины $\mathfrak{T}$ с исходными данными $\mathfrak{m}$.

В теории алгоритмов имеет место *тезис Тьюринга:* все вычислимые функции вычисляются на машинах Тьюринга.

4.3. Универсальная машина Тьюринга

Машина Тьюринга называется *универсальной,* если она может при определённых начальных входных данных вычислить любую функцию, которая вычислима на какой-либо машине Тьюринга.

Иначе говоря, с учётом тезиса Тьюринга можно сказать, что универсальная машина Тьюринга способна вычислить любую вычислимую функцию. Доказано, что универсальная машина Тьюринга существует.

Существование универсальной машины Тьюринга означает, что систему команд любой машины $\mathfrak{T}$ можно интерпретировать двояким образом: либо как описание работы конкретного устройства машины $\mathfrak{T}$, либо как программу для универсальной машины $\mathfrak{U}$. При этом универсальная машина $\mathfrak{U}$ имитирует работу машины $\mathfrak{T}$, тратя несколько шагов своей работы на имитацию каждого шага машины $\mathfrak{T}$.

Для современного инженера, проектирующего систему управления, это обстоятельство вполне естественно. Он хорошо знает, что любой алгоритм управления может быть реализован либо аппаратно — построением соответствующей схемы, либо программно — написанием программы для универсального управляющего компьютера. Однако важно сознавать, что сама идея универсального алгоритмического устройства совершенно не связана с развитием современных технических средств его реализации (электроники,

физики твёрдого тела и т. д.) и является не техническим, а математическим фактом, описываемым в абстрактных математических терминах, не зависящих от технических средств, и к тому же опирающимся на крайне малое количество весьма элементарных исходных понятий. Характерно, что основополагающие работы по теории алгоритмов (и, в частности, работы Тьюринга) появились в 30-х гг. XX в., до создания современных компьютеров.

4.4. Язык Brainfuck

Brainfuck — один из известнейших эзотерических[4] языков программирования, придуманный У. Мюллером[5] в 1993 г. для забавы. Язык имеет восемь команд, каждая из которых записывается одним символом. Исходный код программы на языке Brainfuck представляет собой последовательность этих символов без какого-либо дополнительного синтаксиса.

Машина, которой управляют команды Brainfuck, состоит из упорядоченного набора ячеек и указателя текущей ячейки, напоминающих ленту и головку машины Тьюринга. Кроме того, подразумевается устройство общения с внешним миром (команды «`.`» и «`,`») через поток ввода и поток вывода. Перечислим команды языка Brainfuck:

1) `>` — перейти к следующей ячейке;
2) `<` — перейти к предыдущей ячейке;
3) `+` — увеличить значение текущей ячейки на 1;
4) `-` — уменьшить значение текущей ячейки на 1;
5) `.` — напечатать значение текущей ячейки;
6) `,` — ввести извне значение и сохранить его в текущей ячейке;
7) `[` — если значение текущей ячейки равно нулю, перейти вперёд по тексту программы на команду, следующую за соответствующей командой «`]`» (с учётом вложенности);

[4] от *греч.* ἐσωτερικός — внутренний. Общее свойство, присущее любому эзотерическому языку — текст программы на нём понятен лишь «посвящённому».

[5] Урбан Доминик Мю́ллер (Urban Dominik Müller) — швейцарский программист.

8) `]` — если значение текущей ячейки не равно нулю, перейти назад по тексту программы на команду, следующую за соответствующей командой «`[`» (с учётом вложенности).

Язык Brainfuck можно описать с помощью эквивалентов языка C (табл. 4.1).

Таблица 4.1

Команды языка Brainfuck и их эквиваленты на языке C

Brainfuck	C	Brainfuck	C
`>`	`++ptr;`	`.`	`putchar(*ptr);`
`<`	`--ptr;`	`,`	`*ptr = getchar();`
`+`	`++*ptr;`	`[`	`while (*ptr) {`
`-`	`--*ptr;`	`]`	`}`

Примечание. Предполагается, что переменная `ptr` объявлена как указатель на байт.

Рассмотрим программу на языке Brainfuck, печатающую фразу «Hello World!» (рис. 4.2). В приведённой программе сначала про-

```
++++++++++[>+++++++>++++++++++>+++>+<<<<-]
>++.>+.+++++++..+++.>++.<<+++++++++++++++.>.+++.
------.--------.>+.>.
```

Рис. 4.2. «Hello World!» на языке Brainfuck

изводится подготовка в памяти (начиная с ячейки № 1) массива значений, близких к ASCII-кодам символов, которые необходимо вывести (а именно 70, 100, 30, 10). Для этого используется десятикратное приращение соответствующих ячеек на 7, 10, 3 и 1:

1) `++++++++++` — присвоить значение 10 ячейке № 0 (счётчику цикла);

2) `[` — выполнять в цикле последующие команды, пока значение текущей ячейки (ячейки № 0) больше нуля;

3) `>+++++++` — увеличить значение ячейки № 1 на 7;

4) `>++++++++++` — увеличить значение ячейки № 2 на 10;

5) `>+++` — увеличить значение ячейки № 3 на 3;

6) `>+` — увеличить значение ячейки № 4 на 1;

7) `<<<<-` — вернуться к ячейке № 0 (счетчику) и уменьшить её значение на 1;

8) `]` — вернуться к началу цикла.

Далее осуществляется получение кодов букв и их вывод:

9) `>++.` — получить код буквы «H» (72) из значения 70, хранящегося в ячейке № 1, и вывести его;

10) `>+.` — получить код буквы «e» (101) из значения 100, хранящегося в ячейке № 2, и вывести его;

11) `+++++++..` — получить код буквы «l» (108) из значения 101, хранящегося в ячейке № 2, и вывести его дважды;

12) `+++.` — получить код буквы «o» (111) из значения 108, хранящегося в ячейке № 2, и вывести его;

13) `>++.` — получить код символа «пробел» (32) из значения 30, хранящегося в ячейке № 3, и вывести его;

14) `<<+++++++++++++++.` — получить код буквы «W» (87) из значения 72, хранящегося в ячейке № 1, и вывести его;

15) `>.` — вывести значение ячейки № 2, так как нужный код буквы «o» (111) уже содержится в этой ячейке;

16) `+++.` — получить код буквы «r» (114) из значения 111, хранящегося в ячейке № 2, и вывести его;

17) `------.` — получить код буквы «l» (108) из значения 114, хранящегося в ячейке № 2, и вывести его;

18) `--------.` — получить код буквы «d» (100) из значения 108, хранящегося в ячейке № 2, и вывести его;

19) `>+.` — получить код знака «!» (33) из значения 32, хранящегося в ячейке № 3, и вывести его;

20) `>.` — вывести код перевода строки (10) из ячейки № 4.

Фразу «Hello World!» выводит и следующая программа, представленная на рис. 4.3, которая вычисляет коды нужных символов, изменяя значение одной и той же ячейки. Она в несколько раз длиннее, чем приведённый выше оптимизированный вариант.

```
++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++.++++++++++++++++
++++++++++++.+++++++..+++.-------------------
------------------------------------------------
---------------.++++++++++++++++++++++++++++
++++++++++++++++++++++++.++++++++++++++++++++
++++++.+++.------.--------.-------------------
------------------------------------------------
----.-----------------------.
```

Рис. 4.3. «Hello World!» на языке Brainfuck, неоптимизированная версия

Несмотря на внешнюю примитивность, Brainfuck с бесконечным набором ячеек имеет тьюринговскую полноту. Следовательно, по потенциальным возможностям он не уступает «настоящим» языкам, подобным C, Pascal или Java. Brainfuck подходит для экспериментов по генетическому программированию из-за простоты синтаксиса и лёгкости генерации исходного кода.

В «классическом» Brainfuck, описанном Мюллером, размер ячейки — один байт, количество ячеек — 30 000. В начальном состоянии указатель находится в крайней левой позиции, а все ячейки заполнены нулями. Увеличение и уменьшение значений ячеек происходят по модулю 256. Ввод и вывод также происходят побайтно, с учётом кодировки ASCII (т. е. в результате операции ввода «`,`» символ «1» будет записан в текущую ячейку как число 49, а операция вывода «`.`», совершённая над ячейкой, содержащей число 65, напечатает латинскую букву «A»). В других вариантах языка размер и количество ячеек может быть бóльшими. Есть версии, где значение ячеек

не является целочисленным (ячейки содержат числа с плавающей точкой).

Существует расширение классического языка Brainfuck, позволяющее использовать *процедуры*. Для поддержки этого расширения дополнительно вводятся три новых команды:

1′) `(` — начало объявления процедуры (идентификатором процедуры служит число, хранящееся в текущей ячейке);

2′) `)` — конец объявления процедуры;

3′) `:` — вызов процедуры с идентификатором, равным числу, хранящемуся в текущей ячейке.

Рассмотрим пример программы с процедурами (рис. 4.4).

```
+(>[-]>[-]>[-]<<<)
+(>+++<)
[-]++:[-]+:
```

Рис. 4.4. Пример программы с процедурами на расширенном языке Brainfuck

Сначала описывается процедура 1, очищающая (сбрасывающая в нуль) три следующих ячейки, начиная от текущей:

1) `+(` — увеличить значение текущей ячейки на 1 (до этого было 0). Это значение будет идентификатором процедуры;

2) `>[-]` — очистить 1-ю ячейку справа от текущей;

3) `>[-]` — очистить 2-ю ячейку справа от текущей;

4) `>[-]` — очистить 3-ю ячейку справа от текущей;

5) `<<<` — возвратиться назад на текущую ячейку;

6) `)` — конец процедуры

Далее описывается процедура 2, увеличивающая значение следующей ячейки на 3:

7) `+(` — увеличить значение текущей ячейки на 1 (до этого было 1). Это значение будет идентификатором процедуры;

8) `>` — перейти к следующей ячейке;

9) `+++` — увеличить значение этой ячейки на 3;

10) `<` — возвратиться назад на текущую ячейку;

11) `)` — конец процедуры;

После этого описывается тело программы:

12) `[-]` — очистить текущую ячейку;

13) `++:` — установить в ячейке значение 2 и вызвать процедуру с этим идентификатором — она увеличит значение следующей ячейки на 3;

14) `[-]` — очистить текущую ячейку;

15) `+:` — установить в ячейке значение 1 и вызвать процедуру с этим идентификатором — она очистит следующие 3 ячейки.

4.5. Примеры решения задач

Сегодня машины Тьюринга рассматриваются чаще всего в качестве «распознавателей языков» или, что равносильно, «решателей проблем». Однако Тьюринг рассматривал свою машину как вычислитель функций, значениями которых являются натуральные числа. В их схеме натуральные числа представлялись в *унарной системе счисления,* как блоки из одного и того же символа, и машина вычисляла, изменяя длину блоков или строя новые блоки на ленте. Рассмотрим следующий простой пример. Запись 1^n означает блок из n символов «1».

Пример 4.1. Сложить числа a и b, представленные в унарной системе счисления. Между числами записан разделитель «*».

Р е ш е н и е. По условию задачи необходимо преобразовать слово $1^a{*}1^b$ в слово 1^{a+b}, т. е. удалить разделитель «*» и сдвинуть одно из слагаемых, например, первое, к другому. Это преобразование

осуществляет машина $\mathfrak{T}_+$ с четырьмя состояниями и следующей системой команд (первая команда введена для случая, когда $a = 0$ и исходное слово имеет вид $*1^b$):

$$
\begin{aligned}
q_1* &\to q_0\Lambda R;\\
q_11 &\to q_2\Lambda R;\\
q_21 &\to q_21R;\\
q_2* &\to q_31L;\\
q_31 &\to q_31L;\\
q_3\Lambda &\to q_0\Lambda R.
\end{aligned}
$$

В этой системе команд перечислены не все сочетания состояний машины и символов ленты: опущены те из них, которые при стандартной начальной конфигурации никогда не встретятся. Граф переходов $\mathfrak{T}_+$ приведён на рис. 4.5. Заключительное состояние обозначено двойной окружностью.

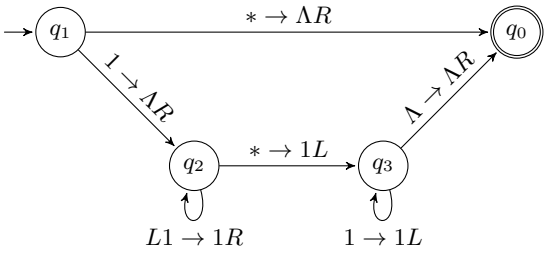

Рис. 4.5. Граф переходов $\mathfrak{T}_+$ для примера 4.1

Заключение

В завершение хотелось бы обозначить некоторые современные направления развития математической логики и теории алгоритмов и их связь с другими дисциплинами, преподаваемыми в рамках специальности 230105 — Программное обеспечение вычислительной техники и автоматизированных систем.

Метод резолюций используется при автоматическом доказательстве теорем. На его основе в 1972 г. был создан язык логического программирования Prolog, изучению которого посвящена часть дисциплины «Функциональное и логическое программирование».

На основе классической логики за последние десятилетия было разработано множество неклассических логик, например, интуиционистская логика, модальные логики, временны́е логики. В настоящее время особую популярность в связи с большой практической значимостью приобрела нечёткая логика. На её основе строятся системы искусственного интеллекта и системы автоматического управления. Нечёткая логика и её применение в советующих системах изучаются в рамках дисциплины «Системы поддержки принятия решений и вычислительного интеллекта».

Помимо рассмотренной в данном пособии машины Тьюринга, существует ряд других абстрактных описаний алгоритмов, эквивалентных между собой. Наиболее важными являются частично-рекурсивные функции, нормальные алгоритмы Маркова и λ-исчисление. Они и их расширения изучаются в рамках таких дисциплин, как «Теория вычислительных процессов», «Теория языков программирования и методы трансляции», «Функциональное и логическое программирование».

Первоначально математическая логика и теория алгоритмов представлялись исключительно как средства обоснования математики. Впоследствии на основе полученных результатов были созданы вычислительные машины и сотни языков программирования, без которых не появилась бы и специальность 230105. Авторы надеются, что выпускники этой специальности будут играть ключевую роль в развитии современного информационного общества, так как оно немыслимо без вычислительной техники и специалистов, знающих фундаментальные принципы её работы.

Библиографический список

1. *Акимов, О. Е.* Дискретная математика: логика, группы, графы / О. Е. Акимов. — М. : Лаборатория Базовых Знаний, 2001. — 352 с. — ISBN 5-93208-056-1.

2. *Босс, В.* Лекции по математике. Т. 6 : От Диофанта до Тьюринга : учеб. пособие / В. Босс. — М. : КомКнига, 2006. — 208 с. — ISBN 5-484-00463-2.

3. *Гильберт, Д.* Основы теоретической логики / Д. Гильберт, В. Аккерман. — М. : ИЛ, 1947. — 304 с.

4. *Гладкий, А. В.* Математическая логика / А. В. Гладкий. — М. : РГГУ, 1998. — 479 с. — ISBN 5-7281-0025-2.

5. *Гуц, А. К.* Математическая логика и теория алгоритмов : учеб. пособие / А. К. Гуц. — Омск : Наследие. Диалог-Сибирь, 2003. — 108 с. — ISBN 5-8239-0126-7.

6. *Игошин, В. И.* Задачи и упражнения по математической логике и теории алгоритмов : учеб. пособие для студ. высш. учеб. заведений / В. И. Игошин. — 4-е изд., стер. — М. : Издательский центр «Академия», 2008. — 304 с. — ISBN 978-5-7695-5272-4.

7. *Игошин, В. И.* Математическая логика и теория алгоритмов : учеб. пособие для студ. высш. учеб. заведений / В. И. Игошин. — 3-е изд., стер. — М. : Издательский центр «Академия», 2008. — 448 с. — ISBN 978-5-7695-4593-1.

8. *Клини, С. К.* Математическая логика / С. К. Клини. — М. : Мир, 1973. — 480 с.

9. *Колмогоров, А. Н.* Математическая логика / А. Н. Колмогоров, А. Г. Драгалин. — Изд. 3-е, стер. — М. : КомКнига, 2006. — 240 с. — ISBN 5-484-00520-5.

10. *Кондаков, Н. И.* Логический словарь-справочник / Н. И. Кондаков. — 2-е изд., испр. и доп. — М. : Наука, 1975. — 720 с.

11. *Лихтарников, Л. М.* Математическая логика. Курс лекций. Задачник-практикум и решения : учеб. пособие. / Л. М. Лихтарников, Т. Г. Сукачева. — 4-е изд., стер. — СПб. : Лань, 2009. — 288 с. — ISBN 978-5-8114-0082-9.

12. *Мальцев, А. И.* Алгоритмы и рекурсивные функции / А. И. Мальцев. — М. : Наука. Гл. ред. физ.-мат. лит., 1986. — 368 с.

13. *Математика.* Большой энциклопедический словарь / гл. ред. Ю. В. Прохоров. — 3-е изд. — М. : Большая Российская Энциклопедия, 2000. — 848 с. — ISBN 5-85270-278-1.

14. *Новиков, П. С.* Элементы математической логики / П. С. Новиков. — 2-е изд., испр. — М. : Наука. Гл. ред. физ.-мат. лит., 1973. — 400 с.

15. *Новиков, Ф. А.* Дискретная математика для программистов : учебник для вузов / Ф. А. Новиков. — 3-е изд. — СПб. : Питер, 2008. — 384 с. — ISBN 978-5-91180-759-7.

16. *Столл, Р.* Множества, логика, аксиоматические теории / Р. Столл. — М. : Просвещение, 1968. — 231 с.

17. *Успенский, В. А.* Вводный курс математической логики / В. А. Успенский, Н. К. Верещагин, В. Е. Плиско. — 2-е изд. — М. : Физматлит, 2004. — 128 с. — ISBN 5-9221-0278-8.

18. *Успенский, В. А.* Машина Поста / В. А. Успенский. — 2-е изд., испр. — М. : Наука. Гл. ред. физ.-мат. лит., 1988. — 96 с. — ISBN 5-02-013735-9.

19. *Формальная логика* / И. Я. Чупахин, А. М. Плотников, К. А. Сергеев и др. ; отв. ред. 1 ч. И. Я. Чупахин ; отв. ред. 2 ч. И. Н. Бродский ; Ленингр. гос. ун-т им. А. А. Жданова. — Л. : Изд-во ЛГУ, 1977. — 357 с.

20. *Хаггарти, Р.* Дискретная математика для программистов / Р. Хаггарти. — М. : Техносфера, 2003. — 320 с. — ISBN 5-94836-016-4.

21. *Хопкрофт, Дж.* Введение в теорию автоматов, языков и вычислений / Дж. Хопкрофт, Р. Мотвани, Дж. Ульман. — 2-е изд. — М. : Вильямс, 2002. — 528 с. — ISBN 5-8459-0261–4.

22. *Чень, Ч.* Математическая логика и автоматическое доказательство теорем / Ч. Чень, Р. Ли ; под ред. С. Ю. Маслова. — М. : Наука. Гл. ред. физ.-мат. лит., 1983. — 360 с.

23. *Чёрч, А.* Введение в математическую логику. Т. 1 / А. Чёрч. — М. : ИЛ, 1960. — 484 с.

24. *Шёнфилд, Дж.* Математическая логика / Дж. Шёнфилд. — М. : Наука, 1975. — 528 с.

25. *Brainfuck* [Электронный ресурс] : Материал из Википедии — свободной энциклопедии : Версия страницы, сохранённая 6 мая 2009 / Авторы Википедии // Википедия, свободная энциклопедия. — Электрон. дан. — Сан-Франциско : Фонд Викимедиа, 2009. — Режим доступа: `http://ru.wikipedia.org/wiki/Brainfuck`, свободный. — Загл. с экрана.

Учебное издание

Куценко Дмитрий Александрович
Терехов Денис Витальевич

Математическая логика и теория алгоритмов
Учебное пособие

Подписано в печать 15.12.09. Формат 60×84/16.
Усл. печ. л. 3,7. Уч.-изд. л. 4,0. Тираж 100 экз.
Заказ № Цена

Отпечатано в Белгородском государственном
технологическом университете им. В. Г. Шухова.
308012, г. Белгород, Костюкова, 46