ФЕДЕРАЛЬНОЕ АГЕНТСТВО ВОЗДУШНОГО ТРАНСПОРТА

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ГРАЖДАНСКОЙ АВИАЦИИ» (МГТУ ГА)

Кафедра высшей математики

А.В. Самохин, Ю.И. Дементьев

МАТЕМАТИЧЕСКАЯ ЛОГИКА

Учебное пособие

Утверждено редакционно-издательским советом МГТУ ГА в качестве учебного пособия

Москва
ИД Академии Жуковского
2018

УДК 510.6(075.8)
ББК 517
С17

Печатается по решению редакционно-издательского совета
Московского государственного технического университета ГА

Рецензенты:
Илларионова О.Г. (МГТУ ГА) – канд. физ.-мат. наук, доц. каф. ВМ;
Красильщик И.С. (ИПУ РАН) – д-р физ.-мат. наук, проф.

Самохин А.В.

С17 Математическая логика [Текст] : учебное пособие / А.В. Самохин, Ю.И. Дементьев. – М. : ИД Академии Жуковского, 2018. – 80 с.

ISBN 978-5-907081-53-6

Данное учебное пособие издаётся в соответствии с рабочей программой учебной дисциплины «Математическая логика» для студентов I курса направления 09.03.01 «Информатика и вычислительная техника» очной формы обучения.

Учебное пособие охватывает разделы математики, изучаемые студентами по дисциплине «Математическая логика»: множества, мощности множеств, эквивалентности и порядок, исчисление высказываний, предикатов, аксиомы и правила вывода, вычислимость, разрешимость, перечислимость, машины Тьюринга и арифметичность.

Рассмотрено и одобрено на заседании кафедры 29.05.2018 г. и методического совета 11.06.2018 г.

УДК 510.6(075.8)
ББК 517
Св. тем. план 2018 г.
поз. 46

САМОХИН Алексей Васильевич, ДЕМЕНТЬЕВ Юрий Игоревич
МАТЕМАТИЧЕСКАЯ ЛОГИКА
Учебное пособие

В авторской редакции

Подписано в печать 29.11.2018 г.
Формат 60x84/16 Печ. л. 5 Усл. печ. л. 4,65
Заказ № 374/1029-УП04 Тираж 35 экз.

Московский государственный технический университет ГА
125993, Москва, Кронштадтский бульвар, д. 20

Издательский дом Академии имени Н. Е. Жуковского
125167, Москва, 8-го Марта 4-я ул., д. 6А
Тел.: (495) 973-45-68 E-mail: zakaz@itsbook.ru

ISBN 978-5-907081-53-6

Оглавление

Предисловие

Имеет ли отношение математическая логика к тому, что необходимо знать специалисту по ЭВМ?

Мы надеемся, что в результате изучения этого курса слушатель убедится, что имеет, и самое непосредственное. Так, глава II — к автоматическому порождению синтаксически правильных текстов, т.е. к специальному программированию; остаток книги посвящен основам теории алгоритмов: здесь обсуждаются, какие задачи вообще являются алгоритмически разрешимыми и какова сложность соответствующих алгоритмов. В главе I собран материал по началам теории множеств, необходимый для понимания остального текста (как, впрочем, и почти всей математики).

В пособии использованы материалы из свободно распространяемых книг [**1**], [**2**] и [**3**] интернет-библиотеки МЦНМО с любезного согласия авторов.

Глава I

Множества и функции

§1. Множества

Основные понятия и обозначения, связанные с множествами и операциями над ними:

- *Множества* состоят из *элементов*. Запись $x \in M$ означает, что x является элементом множества M.
- Говорят, что множество A является *подмножеством* множества B (запись: $A \subset B$), если все элементы A являются элементами B.
- Множества A и B *равны* (запись: $A = B$), если они содержат одни и те же элементы (другими словами, если $A \subset B$ и $B \subset A$).
- Если A — подмножество B, не равное всему B, то A называют *собственным* подмножеством B (запись: $A \subsetneq B$).
- *Пустое* множество $\varnothing$ не содержит ни одного элемента и является подмножеством любого множества.
- *Пересечение* $A \cap B$ двух множеств A и B состоит из элементов, которые принадлежат обоим множествам A и B. Это записывают так:

  $$A \cap B = \{x \mid x \in A \text{ и } x \in B\}$$

  (читается: множество таких x, что $\dots$).
- *Объединение* $A \cup B$ состоит из элементов, которые принадлежат хотя бы одному из множеств A и B:

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

- *Разность* $A \setminus B$ состоит из элементов, которые принадлежат A, но не принадлежат B:

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

  Если множество B является подмножеством множества A, разность $A \setminus B$ называют также *дополнением B до A*.
- *Симметрическая разность* $A \triangle B$ состоит из элементов, которые принадлежат ровно одному из множеств A и B:

  $$A \triangle B = (A \setminus B) \cup (B \setminus A) = (A \cup B) \setminus (A \cap B).$$

- Через $\{a, b, c\}$ обозначается множество, которое содержит элементы a, b, c и не содержит других. Если среди a, b, c есть равные, оно может содержать один или два элемента. Подобное обозначение используется и в менее формальных ситуациях: множество членов последовательности $a_0, a_1, \ldots$ обозначается $\{a_0, a_1, \ldots\}$ или даже $\{a_i\}$. Более аккуратная запись для того же множества такова: $\{a_i \mid i \in \mathbb{N}\}$, где $\mathbb{N}$ — множество натуральных чисел $\{0, 1, 2, \ldots\}$.

Понятие множества появилось в математике сравнительно недавно, в конце 19-го века, в связи с работами Кантора. Отметим,что в школе натуральные числа начинаются с единицы, а в некоторых книжках — с нуля (мы тоже будем называть нуль натуральным числом).

Мы предполагаем, что перечисленные выше основные понятия теории множеств более или менее вам знакомы.

§2. Число элементов

Число элементов в конечном множестве A называют также его *мощностью* и обозначают $|A|$ (а также $\#A$). (Вскоре мы будем говорить о мощностях и для бесконечных множеств.) Следующая формула позволяет найти мощность объединения нескольких множеств, если известны мощности каждого из них, а также мощности всех пересечений.

Теорема 1 (Формула включений и исключений).

$$\begin{aligned} |A \cup B| &= |A| + |B| - |A \cap B|; \\ |A \cup B \cup C| &= |A| + |B| + |C| - \\ &\quad - |A \cap B| - |A \cap C| - |B \cap C| + \\ &\quad + |A \cap B \cap C|; \end{aligned}$$

вообще $|A_1 \cup \ldots \cup A_n|$ *равно*

$$\sum_i |A_i| - \sum_{i<j} |A_i \cap A_j| + \sum_{i<j<k} |A_i \cap A_j \cap A_k| - \ldots$$

Доказательство. Фиксируем произвольное множество U, подмножествами которого являются множества $A_1, \ldots, A_n$.

Характеристической функцией множества $X \subset U$ называют функцию χ_X, которая равна 1 на элементах X и 0 на остальных элементах U. Операции над подмножествами множества U соответствуют операциям с их

характеристическими функциями. В частности, пересечению множеств соответствует произведение характеристических функций:

$$\chi_{A\cap B}(u) = \chi_A(u)\chi_B(u)$$

Дополнению (до U) соответствует функция $1-\chi$, если χ — характеристическая функция исходного множества.

Число элементов множества можно записать как сумму значений его характеристической функции:

$$|X| = \sum_u \chi_X(u).$$

Объединение $A_1 \cup \ldots \cup A_N$ можно записать как дополнение к пересечению дополнений множеств A_i; в терминах характеристических функций имеем

$$\chi_{A_1\cup\ldots\cup A_n} = 1 - (1-\chi_{A_1})\ldots(1-\chi_{A_n}).$$

Раскрыв скобки в правой части, мы получим

$$\sum_i \chi_{A_i} - \sum_{i<j} \chi_{A_i}\chi_{A_j} + \sum_{i<j<k} \chi_{A_i}\chi_{A_j}\chi_{A_k} - \ldots$$

и просуммировав левую и правую часть по всем элементам U (обе они есть функции на U), получим формулу включений и исключений. □

Подсчёт количеств элементов в конечных множествах относят к *комбинаторике*. Сейчас нас в первую очередь интересует следующий принцип:

если между двумя множествами можно установить взаимно однозначное соответствие, то в них одинаковое число элементов.

(Взаимная однозначность требует, чтобы каждому элементу первого множества соответствовал ровно один элемент второго и наоборот.)

§3. Равномощные множества

Два множества называют *равномощными*, если между ними можно установить взаимно однозначное соответствие, при котором каждому элементу одного множества соответствует ровно один элемент другого.

Для конечных множеств это означает, что в них одинаковое число элементов, но определение имеет смысл и для бесконечных множеств. Например, отрезки $[0,1]$ и $[0,2]$ равномощны, поскольку отображение $x \mapsto 2x$ осуществляет искомое соответствие.

Несколько более сложна такая задача: доказать, что интервал $(0,1)$ и луч $(0,+\infty)$ равномощны. Это делается так. Заметим, что отображение $x \mapsto 1/x$ является взаимно однозначным соответствием между $(0,1)$ и $(1,+\infty)$,

а $x \mapsto (x-1)$ — взаимно однозначным соответствием между $(1,+\infty)$ и $(0,+\infty)$, поэтому их композиция $x \mapsto (1/x) - 1$ является искомым взаимно однозначным соответствием между $(0,1)$ и $(0,+\infty)$.

Вообще, как говорят, отношение равномощности есть *отношение эквивалентности*. Это означает, что оно *рефлексивно* (каждое множество равномощно самому себе), *симметрично* (если A равномощно B, то и B равномощно A) и *транзитивно* (если A равномощно B и B равномощно C, то A равномощно C). Свойством транзитивности мы только что воспользовались, взяв луч $(1,+\infty)$ в качестве промежуточного множества.

Ещё несколько примеров:

- Множество бесконечных последовательностей нулей и единиц равномощно множеству всех подмножеств натурального ряда. (В самом деле, сопоставим с каждой последовательностью множество номеров мест, на которых стоят единицы: например, последовательность из одних нулей соответствует пустому множеству, из одних единиц — натуральному ряду, а последовательность $10101010\ldots$ — множеству чётных чисел.)
- Множество бесконечных последовательностей цифр 0, 1, 2, 3 равномощно множеству бесконечных последовательностей цифр 0 и 1. (В самом деле, можно закодировать цифры 0, 1, 2, 3 группами 00, 01, 10, 11. Обратное преобразование разбивает последовательность нулей и единиц на пары, после чего каждая пара заменяется на цифру от 0 до 3.)
- Множество бесконечных последовательностей цифр 0, 1, 2 равномощно множеству бесконечных последовательностей цифр 0 и 1. (Можно было бы пытаться рассуждать так: это множество заключено между двумя множествами одной и той же мощности, и потому равномощно каждому из них. Это верно , но ещё проще закодировать цифры 0, 1 и 2 последовательностями 0, 10 и 11: легко сообразить, что всякая последовательность нулей и единиц однозначно разбивается на такие блоки слева направо. Такой способ кодирования называют префиксным кодом.)
- Пример с последовательностями нулей и единиц можно обобщить: множество подмножеств любого множества U (оно обычно обозначается $P(U)$ и по- английски называется power set) равномощно множеству всех функций, которые ставят в соответствие каждому элементу $x \in U$ одно из чисел 0 и 1 (множество таких функций обычно обозначают 2^X). (В самом деле, каждому множеству $X \subset U$ соответствует его характеристическая функция.)

Мы продолжим этот список, но сначала полезно доказать несколько простых фактов о счётных множествах (равномощных множеству натуральных чисел).

§4. Счётные множества

Множество называется *счётным*, если оно равномощно множеству $\mathbb{N}$ натуральных чисел, то есть если его можно представить в виде $\{x_0, x_1, x_2, \dots\}$ (здесь x_i — элемент, соответствующий числу i; соответствие взаимно однозначно, так что все x_i различны).

Например, множество целых чисел $\mathbb{Z}$ счётно, так как целые числа можно расположить в последовательность $0, 1, -1, 2, -2, 3, -3, \dots$

Теорема 2. *(а) Подмножество счётного множества конечно или счётно.*

(б) Всякое бесконечное множество содержит счётное подмножество.

(в) Объединение конечного или счётного числа конечных или счётных множеств конечно или счётно.

Доказательство. (а) Пусть B — подмножество счётного множества $A = \{a_0, a_1, a_2, \dots\}$. Выбросим из последовательности $a_0, a_1, \dots$ те члены, которые не принадлежат B (сохраняя порядок оставшихся). Тогда оставшиеся члены образуют либо конечную последовательность (и тогда B конечно), либо бесконечную (и тогда B счётно).

(б) Пусть A бесконечно. Тогда оно непусто и содержит некоторый элемент b_0. Будучи бесконечным, множество A не исчерпывается элементом b_0 — возьмём какой-нибудь другой элемент b_1, и т.д. Получится последовательность $b_0, b_1, \dots$; построение не прервётся ни на каком шаге, поскольку A бесконечно. Теперь множество $B = \{b_0, b_1, \dots\}$ и будет искомым счётным подмножеством. (Заметим, что B вовсе не обязано совпадать с A, даже если A счётно.)

(в) Пусть имеется счётное число счётных множеств $A_1, A_2, \dots$ Расположив элементы каждого из них слева направо в последовательность ($A_i = \{a_{i0}, a_{i1}, \dots\}$) и поместив эти последовательности друг под другом, получим таблицу

$$\begin{array}{ccccc} a_{00} & a_{01} & a_{02} & a_{03} & \dots \\ a_{10} & a_{11} & a_{12} & a_{13} & \dots \\ a_{20} & a_{21} & a_{22} & a_{23} & \dots \\ a_{30} & a_{31} & a_{32} & a_{33} & \dots \\ \dots & \dots & \dots & \dots & \dots \end{array}$$

Теперь эту таблицу можно развернуть в последовательность, например, проходя по очереди диагонали:

$$a_{00},\ a_{01}, a_{10},\ a_{02}, a_{11}, a_{20},\ a_{03}, a_{12}, a_{21}, a_{30}, \ldots$$

Если множества A_i не пересекались, то мы получили искомое представление для их объединения. Если пересекались, то из построенной последовательности надо выбросить повторения.

Если множеств конечное число или какие-то из множеств конечны, то в этой конструкции части членов не будет — и останется либо конечное, либо счётное множество. □

Описанный проход по диагоналям задаёт взаимно однозначное соответствие между множеством всех пар натуральных чисел (которое обозначается $\mathbb{N} \times \mathbb{N}$) и $\mathbb{N}$.

Замечание. В доказательстве утверждения (б) теоремы 2 есть тонкий момент: на каждом шаге мы должны выбрать один из оставшихся элементов множества A; такие элементы есть, но у нас нет никакого правила, позволяющего такой выбор описать. При более формальном построении теории множеств тут нужно сослаться на специальную аксиому, называемую *аксиомой выбора*. Законность этой аксиомы вызывала большие споры в начале 20-го века, но постепенно к ней привыкли, и эти споры сейчас почти не воспринимаются.

Ещё несколько примеров счётных множеств:

- Множество $\mathbb{Q}$ рациональных чисел счётно. В самом деле, рациональные числа представляются несократимыми дробями с целым числителем и знаменателем. Множество дробей с данным знаменателем счётно, поэтому $\mathbb{Q}$ представимо в виде объединения счётного числа счётных множеств. Отметим, что множество $\mathbb{R}$ всех действительных чисел несчётно.
- Множество $\mathbb{N}^k$, элементами которого являются наборы из k натуральных чисел, счётно. Это легко доказать индукцией по k. При $k = 2$ множество $\mathbb{N}^2 = \mathbb{N} \times \mathbb{N}$ пар натуральных чисел разбивается на счётное число счётных множеств $\{0\} \times \mathbb{N}, \{1\} \times \mathbb{N}, \ldots$ (элементами i-го множества будут пары, первый член которых равен i). Поэтому $\mathbb{N}^2$ счётно. Аналогичным образом множество $\mathbb{N}^3$ троек натуральных чисел разбивается на счётное число множеств $\{i\} \times \mathbb{N} \times \mathbb{N}$. Каждое из них состоит из троек, первый член которых фиксирован и потому равномощно множеству $\mathbb{N}^2$, которое счётно. Точно так же можно перейти от счётности множества $\mathbb{N}^k$ к счётности множества $\mathbb{N}^{k+1}$.

- Множество всех конечных последовательностей натуральных чисел счётно. В самом деле, множество всех последовательностей данной длины счётно (как мы только что видели), так что интересующее нас множество разбивается на счётное число счётных множеств.
- В предыдущем примере не обязательно говорить о натуральных числах — можно взять любое счётное (или конечное) множество. Например, множество всех текстов, использующих русский алфавит (такой текст можно считать конечной последовательностью букв, пробелов, знаков препинания и т.п.), счётно; ***то же самое можно сказать о множестве (всех мыслимых) компьютерных программ*** и т.д.
- Множество периодических дробей счётно. В самом деле, такая дробь может быть записана как конечная последовательность символов из конечного множества (запятая, цифры, скобки); например, дробь $0{,}16666\ldots$ можно записать как $0{,}1(6)$. А таких последовательностей счётное множество.

Теорема 3. *Если множество A бесконечно, а множество B конечно или счётно, то объединение $A \cup B$ равномощно A.*

Доказательство. Можно считать, что B не пересекается с A (пересечение можно выбросить из B, останется по- прежнему конечное или счётное множество).

Выделим в A счётное подмножество P; остаток обозначим через Q. Тогда нам надо доказать, что $B+P+Q$ равномощно $P+Q$ (знак $+$ символизирует объединение непересекающихся множеств). Поскольку $B+P$ и P оба счётны, между ними существует взаимно однозначное соответствие. Его легко продолжить до соответствия между $B+P+Q$ и $P+Q$ (каждый элемент множества Q соответствует сам себе). □

Теорема 3 показывает, что добавление счётного множества к бесконечному не меняет его мощности. Можно ли сказать то же самое про удаление? Докажите, что если A бесконечно и не является счётным, а B конечно или счётно, то $A \setminus B$ равномощно B.

Добавляя конечные или счётные множества, легко понять, что прямая, все промежутки на прямой (отрезки, интервалы, полуинтервалы), лучи, их конечные или счётные объединения и т.п. равномощны друг другу.

Теорема 4. *Отрезок $[0,1]$ равномощен множеству всех бесконечных последовательностей нулей и единиц.*

Доказательство. В самом деле, каждое число $x \in [0,1]$ записывается в виде бесконечной двоичной дроби. Первый знак этой дроби равен 0 или 1

в зависимости от того, попадает ли число x в левую или правую половину отрезка. Чтобы определить следующий знак, надо выбранную половину поделить снова пополам и посмотреть, куда попадёт x, и т.д.

Это же соответствие можно описать в другую сторону: последовательности $x_0x_1x_2\ldots$ соответствует число, являющееся суммой ряда

$$\frac{x_0}{2}+\frac{x_1}{4}+\frac{x_2}{8}+\ldots$$

(В этом построении мы используем некоторые факты из математического анализа, что не удивительно — нас интересуют свойства действительных чисел.)

Описанное соответствие пока что не совсем взаимно однозначно: двоично-рациональные числа (дроби вида $m/2^n$) имеют два представления. Например, число $3/8$ можно записать как в виде $0{,}011000\ldots$, так и в виде $0{,}010111\ldots$ Соответствие станет взаимно однозначным, если отбросить дроби с единицей в периоде. Но таких дробей счётное число, поэтому на мощность это не повлияет. □

В этом доказательстве можно было бы использовать более привычные десятичные дроби вместо двоичных. Получилось бы, что отрезок $[0,1]$ равномощен множеству всех бесконечных последовательностей цифр $0,1,\ldots,9$. Чтобы перейти отсюда к последовательностям нулей и единиц, можно воспользоваться приёмом, описанным на с. 9.

Теперь всё готово для доказательства такого удивительного факта:

Теорема 5. *Квадрат (со внутренностью) равномощен отрезку.*

Доказательство. Квадрат равномощен множеству $[0,1]\times[0,1]$ пар действительных чисел, каждое из которых лежит на отрезке $[0,1]$ (метод координат). Мы уже знаем, что вместо чисел на отрезке можно говорить о последовательностях нулей и единиц. Осталось заметить, что паре последовательностей нулей и единиц $\langle x_0x_1x_2\ldots, y_0y_1y_2\ldots\rangle$ можно поставить в соответствие последовательность-смесь $x_0y_0x_1y_1x_2y_2\ldots$ и что это соответствие будет взаимно однозначным. □

Из теоремы 5 легко получить много других утверждений про равномощность геометрических объектов: круг равномощен окружности, прямая равномощна плоскости и т.п.

Можно также заметить, что пространство (точки которого задаются тремя координатами $\langle x,y,z\rangle$) равномощно плоскости (надо закодировать пару $\langle x,y\rangle$ одним числом), и, следовательно, прямой. То же самое можно проделать и для пространств большей размерности.

Отметим, что мы пока не умеем доказывать, что множество действительных чисел (или множество бесконечных последовательностей нулей и единиц) несчётно.

Мощность множества действительных чисел называют *мощностью континуума* (от латинского слова, означающего непрерывный; имеется в виду, что точка на отрезке может непрерывно двигаться от одного конца к другому).

§5. Теоремы Кантора-Бернштейна и Кантора

Определение равномощности уточняет интуитивную идею о множествах одинакового размера. А как формально определить, когда одно множество больше другого?

Говорят, что множество A *по мощности не больше* множества B, если оно равномощно некоторому подмножеству множества B (возможно, самому B).

Отношение иметь не большую мощность обладает многими естественными свойствами:

- Если A и B равномощны, то A имеет не большую мощность, чем B. (Очевидно.)
- Если A имеет не большую мощность, чем B, а B имеет не большую мощность, чем C, то A имеет не большую мощность, чем C. (Тоже несложно. Пусть A находится во взаимно однозначном соответствии с $B' \subset B$, а B находится во взаимно однозначном соответствии с $C' \subset C$. Тогда при втором соответствии B' соответствует некоторому множеству $C'' \subset C' \subset C$, и потому A равномощно C''.)
- Если A имеет не большую мощность, чем B, а B имеет не большую мощность, чем A, то они равномощны. (Это вовсе не очевидное утверждение составляет содержание теоремы Кантора-Бернштейна, которую мы сейчас докажем.)
- Для любых двух множеств A и B верно (хотя бы) одно из двух: либо A имеет не большую мощность, чем B, либо B имеет не большую мощность, чем A. (Доказательство этого факта требует так называемой трансфинитной индукции и здесь принимается на веру)

Теорема 6 (Кантора-Бернштейна). *Если множество A равномощно некоторому подмножеству множества B, а B равномощно некоторому подмножеству множества A, то множества A и B равномощны.*

Доказательство несложное, но длинное, и мы его опустим.

Теорема Кантора - Бернштейна значительно упрощает доказательства равномощности: например, если мы хотим доказать, что бублик и шар в пространстве равномощны, то достаточно заметить, что из бублика можно вырезать маленький шар (гомотетичный большому), а из шара — маленький бублик.

Теперь, имея в виду теорему Кантора-Бернштейна, вернёмся к вопросу о сравнении мощностей. Для данных множеств A и B теоретически имеются четыре возможности:

- A равномощно некоторой части B, а B равномощно некоторой части A. (В этом случае, как мы знаем, множества равномощны.)
- A равномощно некоторой части B, но B не равномощно никакой части A. В этом случае говорят, что A *имеет меньшую мощность, чем* B.
- B равномощно некоторой части A, но A не равномощно никакой части B. В этом случае говорят, что A *имеет большую мощность, чем* B.
- Ни A не равномощно никакой части B, ни B не равномощно никакой части A. (Этот случай на самом деле невозможен).

Заметим, что мы уже долго говорим о сравнении мощностей, но воздерживаемся от упоминания мощности множества как самостоятельного объекта, а только сравниваем мощности разных множеств.

Так или иначе, мы будем употреблять обозначение $|A|$ для мощности множества A хотя бы как вольность речи: $|A| = |B|$ означает, что множества A и B равномощны; $|A| \leq |B|$ означает, что A равномощно некоторому подмножеству множества B, а $|A| < |B|$ означает, что A имеет меньшую мощность, чем B (см. с. 15).

Классический пример неравномощных бесконечных множеств (до сих пор такого примера у нас не было!) даёт диагональная конструкция Кантора.

Теорема 7 (Кантора). *Множество бесконечных последовательностей нулей и единиц несчётно.*

Доказательство. Предположим, что оно счётно. Тогда все последовательности нулей и единиц можно перенумеровать: $\alpha_0, \alpha_1, \ldots$ Составим бесконечную вниз таблицу, строками которой будут наши последовательности:

$$\begin{array}{lcllll} \alpha_0 & = & \underline{\alpha_{00}} & \alpha_{01} & \alpha_{02} & \ldots \\ \alpha_1 & = & \alpha_{10} & \underline{\alpha_{11}} & \alpha_{12} & \ldots \\ \alpha_2 & = & \alpha_{20} & \alpha_{21} & \underline{\alpha_{22}} & \ldots \\ & & \multicolumn{3}{c}{\ldots\ldots\ldots\ldots} \end{array}$$

(через α_{ij} мы обозначаем j-й член i- й последовательности). Теперь рассмотрим последовательность, образованную стоящими на диагонали членами α_{00}, α_{11}, α_{22}, ...; её i-й член есть α_{ii} и совпадает с i-м членом i-й последовательности. Заменив все члены на противоположные, мы получим последовательность β, у которой

$$\beta_i = 1 - \alpha_{ii},$$

так что последовательность β отличается от любой из последовательностей α_i (в позиции i) и потому отсутствует в таблице. А мы предположили, что таблица включает в себя все последовательности — противоречие. □

Из этой теоремы следует, что множество $\mathbb{R}$ действительных чисел (которое, как мы видели, равномощно множеству последовательностей нулей и единиц) несчётно.

Вернёмся к диагональной конструкции. Мы знаем, что множество последовательностей нулей и единиц равномощно множеству подмножеств натурального ряда (каждому подмножеству соответствует его характеристическая последовательность, у которой единицы стоят на местах из этого подмножества). Поэтому можно переформулировать эту теорему так:

Множество $\mathbb{N}$ не равномощно множеству своих подмножеств.

Доказательство также можно шаг за шагом перевести на такой язык: пусть они равномощны; тогда есть взаимно однозначное соответствие $i \mapsto A_i$ между натуральными числами и подмножествами натурального ряда. Диагональная последовательность в этих терминах представляет собой множество тех i, для которых $i \in A_i$, а последовательность β, отсутствовавшая в перечислении, теперь будет его дополнением ($B = \{i \mid i \notin A_i\}$).

При этом оказывается несущественным, что мы имеем дело с натуральным рядом, и мы приходим к такому утверждению:

Теорема 8 (общая формулировка теоремы Кантора). *Никакое множество X не равномощно множеству всех своих подмножеств.*

Доказательство. Пусть φ — взаимно однозначное соответствие между X и множеством $P(X)$ всех подмножеств множества X. Рассмотрим те элементы $x \in X$, которые не принадлежат соответствующему им подмножеству. Пусть Z — образованное ими множество:

$$Z = \{x \in X \mid x \notin \varphi(x)\}.$$

Докажем, что подмножество Z не соответствует никакому элементу множества X. Пусть это не так и $Z = \varphi(z)$ для некоторого элемента $z \in X$. Тогда

$$z \in Z \Leftrightarrow z \notin \varphi(z) \Leftrightarrow z \notin Z$$

(первое — по построению множества Z, второе — по предположению $\varphi(z) = Z$). Полученное противоречие показывает, что Z действительно ничему не соответствует, так что φ не взаимно однозначно. □

С другой, стороны, любое множество X равномощно некоторой части множества $P(X)$. В самом деле, каждому элементу $x \in X$ можно поставить в соответствие одноэлементное подмножество $\{x\}$. Поэтому, вспоминая определение сравнения множеств по мощности (с. 15), можно сказать, что мощность множества X всегда меньше мощности множества $P(X)$

На самом деле мы уже приблизились к опасной границе, когда наглядные представления о множествах приводят к противоречию. В самом деле, рассмотрим множество всех множеств U, элементами которого являются все множества. Тогда, в частности, все подмножества множества U будут его элементами, и $P(U) \subset U$, что невозможно по теореме Кантора.

Это рассуждение можно развернуть, вспомнив доказательство теоремы Кантора — получится так называемый парадокс Рассела. Вот как его обычно излагают.

Типичные множества не являются своими элементами. Скажем, множество натуральных чисел $\mathbb{N}$ само не является натуральным числом и потому не будет своим элементом. Однако в принципе можно себе представить и множество, которое является своим элементом (например, множество всех множеств). Назовём такие множества необычными. Рассмотрим теперь множество всех обычных множеств. Будет ли оно обычным? Если оно обычное, то оно является своим элементом и потому необычное, и наоборот. Как же так?

Модифицированная версия этого парадокса такова: будем называть прилагательное самоприменимым, если оно обладает описываемым свойством. Например, прилагательное русский самоприменимо, а прилагательное глиняный нет. Другой пример: прилагательное трёхсложный самоприменимо, а двусложный нет. Теперь вопрос: будет ли прилагательное несамоприменимый самоприменимым? (Любой ответ очевидно приводит к противоречию.)

Отсюда недалеко до широко известного парадокса лжеца, говорящего я лгу, или до истории о солдате, который должен был брить всех солдат одной с ним части, кто не бреется сам и т.п.

Возвращаясь к теории множеств, мы обязаны дать себе отчёт в том, что плохого было в рассуждениях, приведших к парадоксу Рассела. Вопрос этот далеко не простой, и его обсуждение активно шло всю первую половину 20-го века. Итоги этого обсуждения приблизительно можно сформулировать так:

- Понятие множества не является непосредственно очевидным; разные люди (и научные традиции) могут понимать его по-разному.
- Множества — слишком абстрактные объекты для того, чтобы вопрос а что на самом деле? имел смысл.

Подобно тому как вопрос: евклидова или неевклидова геометрия правильна на самом деле, если вообще имеет смысл, не является математическим — скорее об этом следует спрашивать физиков. К теории множеств это относится в ещё большей степени, и разве что теология (Кантор неоднократно обсуждал вопросы теории множеств с профессионалами-теологами) могла бы в принципе претендовать на окончательный ответ.

§6. Функции

До сих пор мы старались ограничиваться минимумом формальностей и говорили о функциях, их аргументах, значениях, композиции и т.п. без попыток дать определения этих понятий. Сейчас мы дадим формальные определения.

Пусть A и B — два множества. Рассмотрим множество всех упорядоченных пар $\langle a, b\rangle$, где $a \in A$ и $b \in B$. Это множество называется *декартовым произведением* множеств A и B и обозначается $A \times B$.

Любое подмножество R множества $A \times B$ называется *отношением* между множествами A и B. Если $A = B$, говорят о *бинарном отношении* на множестве A. Например, на множестве натуральных чисел можно рассмотреть бинарное отношение быть делителем, обычно обозначаемое символом $|$. Тогда можно в принципе было бы написать $\langle 2, 6\rangle \in |$ и $\langle 2, 7\rangle \notin |$. Обычно, однако, знак отношения пишут между объектами (например, $2|6$).

Задача 1. *Вопрос для самоконтроля: отношения быть делителем и делиться на — это одно и то же отношение или разные? (Ответ: конечно, разные — в упорядоченной паре порядок существен.)*

Если аргументами функции являются элементы множества A, а значениями — элементы множества B, то можно рассмотреть отношение между A и B, состоящее из пар вида $\langle x, f(x)\rangle$. По аналогии с графиками функций на плоскости такое множество можно назвать графиком функции f. С формальной точки зрения, однако, удобнее не вводить отдельного неопределяемого понятия функции, а вместо этого отождествить функцию с её графиком.

Отношение $F \subset A \times B$ называется *функцией из A в B*, если оно не содержит пар с одинаковым первым членом и разными вторыми. Другими

словами, это означает, что для каждого $a \in A$ существует не более одного $b \in B$, при котором $\langle a, b\rangle \in F$.

Те элементы $a \in A$, для которых такое b существует, образуют *область определения* функции F. Она обозначается $D(F)$ (от английского слова domain). Для любого элемента $a \in D(F)$ можно определить *значение* функции F на аргументе a (в точке a, как иногда говорят) как тот единственный элемент $b \in B$, для которого $\langle a, b\rangle \in F$. Этот элемент записывают как $F(a)$. Все такие элементы b образуют *множество значений* функции F, которое обозначается $\Im(F)$.

Если $a \notin D(F)$, то говорят, что функция *не определена* на a. Заметим, что по нашему определению функция из A в B не обязана быть определена на всех элементах множества A — её область определения может быть любым подмножеством множества A. Симметричным образом множество её значений может не совпадать с множеством B.

Если область определения функции f из A в B совпадает с A, то пишут $f\colon A \to B$.

Пример: *тождественная* функция $\mathrm{id}_A\colon A \to A$ переводит множество A в себя, причём $\mathrm{id}(a) = a$ для любого $a \in A$. Она представляет собой множество пар вида $\langle a, a\rangle$ для всех $a \in A$. (Индекс A в id_A иногда опускают, если ясно, о каком множестве идёт речь.)

Композицией двух функций $f\colon A \to B$ и $g\colon B \to C$ называют функцию $h\colon A \to C$, определённую соотношением $h(x) = g(f(x))$. Другими словами, h представляет собой множество пар

$$\{\langle a, c\rangle \mid \langle a, b\rangle \in f \text{ и } \langle b, c\rangle \in g \text{ для некоторого } b \in B\}.$$

Композиция функций обозначается $g \circ f$ (мы, как и в большинстве книг, пишем справа функцию, которая применяется первой).

Очевидно, композиция (как операция над функциями) ассоциативна, то есть $h\circ(f\circ g) = (h\circ f)\circ g$, поэтому в композиции нескольких подряд идущих функций можно опускать скобки.

Пусть $f\colon A \to B$. *Прообразом* подмножества $B' \subset B$ называется множество всех элементов $x \in A$, для которых $f(x) \in B'$. Оно обозначается $f^{-1}(B')$:

$$f^{-1}(B') = \{x \in A \mid f(x) \in B'\}.$$

Образом множества $A' \subset A$ называется множество всех значений функции f на всех элементах множества A'. Оно обозначается $f(A')$:

$$\begin{aligned} f(A') &= \{f(a) \mid a \in A'\} = \\ &= \{b \in B \mid \langle a, b\rangle \in f \text{ для некоторого } a \in A'\}. \end{aligned}$$

Иногда вместо функций говорят об отображениях (резервируя термин функция для отображений с числовыми аргументами и значениями). Мы не будем строго придерживаться таких различий, употребляя слова отображение и функция как синонимы.

Функция $f\colon A \to B$ называется *инъективной*, или *инъекцией*, или *вложением*, если она переводит разные элементы в разные, то есть если $f(a_1) \neq f(a_2)$ при различных a_1 и a_2.

Функция $f\colon A \to B$ называется *сюръективной*, или *сюръекцией*, или *наложением*, если множество её значений есть всё B. (Иногда такие функции называют *отображениями на* B.)

Эти два определения более симметричны, чем может показаться на первый взгляд.

Отображение (функция) $f\colon A \to B$, которое одновременно является инъекцией и сюръекцией (вложением и наложением), называется *биекцией*, или взаимно однозначным соответствием.

Если f — биекция, то существует *обратная* функция f^{-1}, для которой $f^{-1}(y) = x \Leftrightarrow f(x) = y$.

Напомним, что множества A и B равномощны, если существует биекция $f\colon A \to B$. В каком случае существует инъекция (вложение) $f\colon A \to B$? Легко понять, что вложение является взаимно однозначным соответствием между A и некоторым подмножеством множества B, поэтому такое вложение существует тогда и только тогда, когда в B есть подмножество, равномощное A, —е когда мощность A не превосходит мощности B (в смысле определения, данного в разделе 5).

Чуть менее очевиден другой результат: наложение A на B существует тогда и только тогда, когда мощность B не превосходит мощности A.

В самом деле, пусть наложение $f\colon A \to B$ существует. Для каждого элемента $b \in B$ найдётся хотя бы один элемент $a \in A$, для которого $f(a) = b$. Выбрав по одному такому элементу, мы получим подмножество $A' \subset A$, которое находится во взаимно однозначном соответствии с множеством B. (Здесь снова используется аксиома выбора, о которой мы говорили на с. 11.)

В обратную сторону: если какое-то подмножество A' множества A равномощно множеству B и имеется биекция $g\colon A' \to B$, то наложение A на B можно получить, доопределив эту биекцию на элементах вне A' каким угодно образом.

На самом деле такое продолжение возможно, только если B непусто, так что правильное утверждение звучит так: наложение A на B существует только и только тогда, когда B непусто и равномощно некоторому подмножеству A, или когда оба множества пусты.

В нашем изложении остаётся ещё один не вполне понятный момент: что такое упорядоченная пара? Неформально говоря, это способ из двух объектов x и y образовать один объект $\langle x, y\rangle$, причём этот способ обладает таким свойством:

$$\langle x_1, y_1\rangle = \langle x_2, y_2\rangle \;\Leftrightarrow\; x_1 = x_2 \text{ и } y_1 = y_2.$$

В принципе, можно так и считать понятие упорядоченной пары неопределяемым, а это свойство — аксиомой.

§7. Отношения эквивалентности и порядка

Напомним, что бинарным отношением на множестве X называется подмножество $R \subset X \times X$; вместо $\langle x_1, x_2\rangle \in R$ часто пишут $x_1 R x_2$.

Бинарное отношение R на множестве X называется *отношением эквивалентности*, если выполнены следующие свойства:

- (рефлексивность) xRx для всех $x \in X$;
- (симметричность) $xRy \Rightarrow yRx$ для всех $x, y \in X$;
- (транзитивность) xRy и $yRz \Rightarrow xRz$ для любых элементов $x, y, z \in X$.

Имеет место следующее очевидное, но часто используемое утверждение:

Теорема 9. *(а) Если множество X разбито в объединение непересекающихся подмножеств, то отношение лежать в одном подмножестве является отношением эквивалентности.*

(б) Всякое отношение эквивалентности получается описанным способом из некоторого разбиения.

Доказательство. Первое утверждение совсем очевидно; мы приведём доказательство второго, чтобы было видно, где используются все пункты определения эквивалентности. Итак, пусть R — отношение эквивалентности. Для каждого элемента $x \in X$ рассмотрим его *класс эквивалентности* — множество всех $y \in X$, для которых верно xRy.

Докажем, что для двух различных x_1, x_2 такие множества либо не пересекаются, либо совпадают. Пусть они пересекаются, то есть имеют общий элемент z. Тогда $x_1 Rz$ и $x_2 Rz$, откуда zRx_2 (симметричность) и $x_1 Rx_2$ (транзитивность), а также $x_2 Rx_1$ (симметричность). Поэтому для любого z из $x_1 Rz$ следует $x_2 Rz$ (транзитивность) и наоборот.

Осталось заметить, что в силу рефлексивности каждый элемент x принадлежит задаваемому им классу, то есть действительно всё множество X разбито на непересекающиеся классы. □

Множество классов эквивалентности называют *фактор- множеством* множества X по отношению эквивалентности R. (Если отношение согласовано с дополнительными структурами на X, получаются фактор-группы, фактор-кольца и т.д.)

Отношения эквивалентности нам не раз ещё встретятся, но сейчас наша основная тема — отношения порядка.

Бинарное отношение $\leq$ на множестве X называется *отношением частичного порядка*, если выполнены такие свойства:

- (рефлексивность) $x \leq x$ для всех $x \in X$;
- (антисимметричность) $x \leq y$ и $y \leq x \Rightarrow x = y$ для всех $x, y \in X$;
- (транзитивность) $x \leq y$ и $y \leq z \Rightarrow x \leq z$ для всех $x, y, z \in X$.

(Следуя традиции, мы используем символ $\leq$ (а не букву) как знак отношения порядка.) Множество с заданным на нём отношением частичного порядка называют *частично упорядоченным*.

Говорят, что два элемента x, y частично упорядоченного множества *сравнимы*, если $x \leq y$ или $y \leq x$. Заметим, что определение частичного порядка не требует, чтобы любые два элемента множества были сравнимы. Добавив это требование, мы получим определение *линейного порядка* (*линейно упорядоченного множества*).

Приведём несколько примеров частичных порядков:

- Числовые множества с обычным отношением порядка (здесь порядок будет линейным).
- На множестве $\mathbb{R} \times \mathbb{R}$ всех пар действительных чисел можно ввести частичный порядок, считая, что $\langle x_1, x_2 \rangle \leq \langle y_1, y_2 \rangle$, если $x_1 \leq x_2$ и $y_1 \leq y_2$. Этот порядок уже не будет линейным: пары $\langle 0, 1 \rangle$ и $\langle 1, 0 \rangle$ не сравнимы.
- На множестве функций с действительными аргументами и значениями можно ввести частичный порядок, считая, что $f \leq g$, если $f(x) \leq g(x)$ при всех $x \in \mathbb{R}$. Этот порядок не будет линейным.
- На множестве целых положительных чисел можно определить порядок, считая, что $x \leq y$, если x делит y. Этот порядок тоже не будет линейным.
- Отношение любой простой делитель числа x является также и делителем числа y не будет отношением порядка на множестве целых положительных чисел (оно рефлексивно и транзитивно, но не антисимметрично).
- Пусть U — произвольное множество. Тогда на множестве $P(U)$ всех подмножеств множества U отношение включения $\subset$ будет частичным порядком.

- На буквах русского алфавита традиция определяет некоторый порядок (а $\le$ б $\le$ в $\le \ldots \le$ я). Этот порядок линеен — про любые две буквы можно сказать, какая из них раньше (при необходимости заглянув в словарь).
- На словах русского алфавита определён *лексикографический* порядок (как в словаре). Формально определить его можно так: если слово x является началом слова y, то $x \le y$ (например, кант $\le$ кантор). Если ни одно из слов не является началом другого, посмотрим на первую по порядку букву, в которой слова отличаются: то слово, где эта буква меньше в алфавитном порядке, и будет меньше. Этот порядок также линеен (иначе что бы делали составители словарей?).
- Отношение равенства ($(x \le y) \Leftrightarrow (x = y)$) также является отношением частичного порядка, для которого никакие два различных элемента не сравнимы.
- Приведём теперь бытовой пример. Пусть есть множество X картонных коробок. Введём на нём порядок, считая, что $x \le y$, если коробка x целиком помещается внутрь коробки y (или если x и y — одна и та же коробка). В зависимости от набора коробок этот порядок может быть или не быть линейным.

Пусть x, y — элементы частично упорядоченного множества X. Говорят, что $x < y$, если $x \le y$ и $x \ne y$. Для этого отношения выполнены такие свойства:

$$x \not< x;$$
$$(x < y) \text{ и } (y < z) \;\Rightarrow\; x < z.$$

(Первое очевидно, проверим второе: если $x < y$ и $y < z$, то есть $x \le y$, $x \ne y$, $y \le z$, $y \ne z$, то $x \le z$ по транзитивности; если бы оказалось, что $x = z$, то мы бы имели $x \le y \le x$ и потому $x = y$ по антисимметричности, что противоречит предположению.)

Терминологическое замечание: мы читаем знак $\le$ как меньше или равно, а знак $<$ — как меньше, неявно предполагая, что $x \le y$ тогда и только тогда, когда $x < y$ или $x = y$. К счастью, это действительно так. Ещё одно замечание: выражение $x > y$ (x больше y) означает, что $y < x$, а выражение $x \ge y$ (x больше или равно y) означает, что $y \le x$.

В некоторых книжках отношение частичного порядка определяется как отношение $<$, удовлетворяющее двум указанным свойствам. В этом случае отношение $x \le y \Leftrightarrow [(x < y) \text{ или } (x = y)]$ является отношением частичного порядка в смысле нашего определения.

Во избежание путаницы отношение $<$ иногда называют отношением *строгого порядка*, а отношение $\leq$ — отношением *нестрогого порядка*. Одно и то же частично упорядоченное множество можно задавать по-разному: можно сначала определить отношение нестрогого порядка $\leq$ (рефлексивное, антисимметричное и транзитивное) и затем из него получить отношение строгого порядка $<$, а можно действовать и наоборот.

Элемент частично упорядоченного множества называют *наибольшим*, если он больше любого другого элемента, и *максимальным*, если не существует большего элемента. Если множество не является линейно упорядоченным, то это не одно и то же: наибольший элемент автоматически является максимальным, но не наоборот. (Одно дело коробка, в которую помещается любая другая, другое — коробка, которая никуда больше не помещается.)

Аналогичным образом определяются *наименьшие* и *минимальные* элементы.

Легко понять, что наибольший элемент в данном частично упорядоченном множестве может быть только один, в то время как максимальных элементов может быть много.

Глава II

Математическая логика

§1. Высказывания и операции

"Если число π рационально, то π — алгебраическое число. Но оно не алгебраическое. Значит, π не рационально." Мы не обязаны знать, что такое число π, какие числа называют рациональными и какие алгебраическим, чтобы признать, что это рассуждение правильно — в том смысле, что из двух сформулированных посылок действительно вытекает заключение. Такого рода ситуации — когда некоторое утверждение верно независимо от смысла входящий в него высказываний — составляют предмет *логики высказываний*.

Наши рассмотрения будут иметь вполне точный математический характер, хотя мы начнём с неформальных мотивировок.

Высказывания могут быть *истинными* и *ложными*. Например, "$2^{16}+1$ — простое число— истинное высказывание, а "$2^{32}+1$ — простое число— ложное (это число делится на 641). Про высказывание "существует бесконечно много простых p, для которых $p+2$ — также простое" никто не берётся сказать наверняка, истинно оно или ложно. Заметим, что "x делится на 2" в этом смысле не является высказыванием, пока не сказано, чему равно x; при разных x получаются разные высказывания, одни истинные (при чётном x), другие — ложные (при нечётном x).

Высказывания можно соединять друг с другом с помощью "логических связок". Эти связки имеют довольно странные, но традиционные названия и обозначения (табл. 1). Отметим также, что в импликации $A \Rightarrow B$ высказывание A называют *посылкой*, или *антецедентом импликации*, а B — *заключением*, или *консеквентом*.

Говорят также, что высказывание имеет *истинностное значение* **И** (истина), если оно истинно, или **Л** (ложь), если оно ложно. Иногда вместо **И** употребляется буква **Т** (true) или число 1, а вместо **Л** — буква **F** (false) или число 0. (С первого взгляда идея произвольным образом выбрать числа 0 и 1 кажется дикой — какая бы польза могла быть от, скажем, сложения истинностных значений? Удивительным образом в последние годы обнаружилось,

связка	обозначение	название
A и B	$A\&B$ $A \wedge B$ A `and` B	конъюнкция
A или B	$A \vee B$ A `or` B	дизъюнкция
не A A неверно	$\neg A$ $\sim A$ $\overline{A}$ `not` A	отрицание
из A следует B если A, то B A влечёт B B — следствие A	$A \to B$ $A \Rightarrow B$ $A \supset B$ `if` A `then` B	импликация следование

Таблица 1. Логические связки, обозначения и названия.

что такая польза есть, и если оперировать с истиной и ложью как элементами конечного поля, можно получить много неожиданных результатов. Но это выходит за рамки нашей книги.)

Логические связки позволяют составлять сложные высказывания из простых. При этом истинность составного высказывания определяется истинностью его частей в соответствии с таблицей 2.

A	B	$A \wedge B$	$A \vee B$	$A \to B$
Л	**Л**	**Л**	**Л**	**И**
Л	**И**	**Л**	**И**	**И**
И	**Л**	**Л**	**И**	**Л**
И	**И**	**И**	**И**	**И**

A	$\neg A$
Л	**И**
И	**Л**

Таблица 2. Таблицы истинности для логических связок.

Те же правила можно изложить словесно. Высказывание $A \wedge B$ истинно, если оба высказывания A и B истинны. Высказывание $A \vee B$ истинно, если хотя бы одно из высказываний A и B истинно. Высказывание $A \to B$ ложно в единственном случае: если A истинно, а B ложно. Наконец, $\neg A$ истинно в том и только том случае, когда A ложно.

Из всех связок больше всего вопросов вызывает импликация. В самом деле, не очень понятно, почему надо считать, скажем, высказывания "если $2 \times 2 = 5$, то $2 \times 2 = 4$" и "если $2 \times 2 = 5$, то $3 \times 3 = 1$" истинными. (Именно

так говорят наши таблицы: **Л** $\to$ **И** = **Л** $\to$ **Л** = **И**.) Следующий пример показывает, что в таком определении есть смысл.

Общепризнано, что если число x делится на 4, то оно делится на 2. Это означает, что высказывание

$$(x \text{ делится на } 4) \ \to \ (x \text{ делится на } 2)$$

истинно при всех x. Подставим сюда $x = 5$: обе части ложны, а утверждение в целом истинно. При $x = 6$ посылка импликации ложна, а заключение истинно, и вся импликация истинна. Наконец, при $x = 8$ посылка и заключение истинны и импликация в целом истинна. С другой стороны, обратное утверждение (если x делится на 2, то x делится на 4) неверно, и число 2 является контрпримером. При этом посылка импликации истинна, заключение ложно, и сама импликация ложна. Таким образом, если считать, что истинность импликации определяется истинностью её частей (а не наличием между ними каких-то причинно-следственных связей), то все строки таблицы истинности обоснованы. Чтобы подчеркнуть такое узко-формальное понимание импликации, философски настроенные логики называют её "материальной импликацией".

Теперь от неформальных разговоров перейдём к определениям. Элементарные высказывания (из которых составляются более сложные) мы будем обозначать маленькими латинскими буквами и называть *пропозициональными переменными*. Из них строятся *пропозициональные формулы* по таким правилам:

- Всякая пропозициональная переменная есть формула.
- Если A — пропозициональная формула, то $\neg A$ — пропозициональная формула.
- Если A и B — пропозициональные формулы, то $(A \wedge B)$, $(A \vee B)$ и $(A \to B)$ — пропозициональные формулы.

Можно ещё сказать так: формулы образуют минимальное множество, обладающее указанными свойствами (слово "минимальное" здесь существенно: ведь если бы мы объявили любую последовательность переменных, скобок и связок формулой, то эти три свойства были бы тоже выполнены).

Пусть формула φ содержит n пропозициональных переменных $p_1, p_2, \ldots, p_n$. Если подставить вместо этих переменных истинностные значения (**И** или **Л**), то по таблицам можно вычислить истинностное значение формулы в целом. Таким образом, формула задаёт некоторую функцию от n аргументов, каждый из которых может принимать значения **Л** и **И**. Значения функции также лежат в множестве {**Л**,**И**}, которое мы будем обозначать $\mathbb{B}$. Мы будем следовать уже упоминавшейся традиции и отождествлять **И** с единицей, а

Л — с нулём, тем самым $\mathbb{B}$ есть $\{0,1\}$. Формула φ задаёт отображение типа $\mathbb{B}^n \to \mathbb{B}$. Такие отображения называют также *булевыми функциями n аргументов*.

Пример. Рассмотрим формулу $(p \wedge (q \wedge \neg r))$. Она истинна в единственном случае — когда p и q истинны, а r ложно (см. таблицу 3).

p	q	r	$\neg r$	$(q \wedge \neg r)$	$(p \wedge (q \wedge \neg r))$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	1	0
0	1	1	0	0	0
1	0	0	1	0	0
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	0	0

Таблица 3. Таблица истинности для $(p \wedge (q \wedge \neg r))$.

Некоторые формулы выражают логические законы — составные высказывания, истинные независимо от смысла их частей. Такие формулы (истинные при всех значениях входящих в них переменных) называют *тавтологиями*.

Пример. Формула $((p \wedge q) \to p)$ является тавтологией (это можно проверить, например, составив таблицу). Она выражает такой логический закон: из конъюнкции утверждений следует первое из них.

Задача 2. *Как выглядит симметричное утверждение для дизъюнкции и какая формула его выражает?*

Две формулы называют *эквивалентными*, если они истинны при одних и тех же значениях переменных (другими словами, если они задают одну и ту же булеву функцию). Например, формула $(p \wedge (p \to q))$ истинна лишь при $p = q =$ **И**, и потому эквивалентна формуле $(p \wedge q)$.

Рассмотрим формулу $((p \wedge q) \vee q)$. Она истинна, если переменная q истинна, и ложна, если переменная q ложна. Хотелось бы сказать, что она эквивалентна формуле q, но тут есть формальная трудность: она содержит две переменные и потому задаёт функцию от двух аргументов (типа $\mathbb{B} \times \mathbb{B} \to \mathbb{B}$), в то время как формула q задаёт функцию одного аргумента. Мы не будем обращать на это внимания и будем считать эти формулы эквивалентными. Вообще, если есть список переменных $p_1, \ldots, p_n$, содержащий все переменные некоторой формулы φ (и, возможно, ещё какие-то переменные), можно

считать, что формула φ задаёт функцию от n аргументов, возможно, на деле зависящую не от всех аргументов (постоянную по некоторым аргументам)

После сделанных оговорок легко проверить следующий факт: формулы φ и ψ эквивалентны тогда и только тогда, когда формула $((\varphi \to \psi) \wedge (\psi \to \varphi))$ является тавтологией. Используя сокращение $(p \leftrightarrow q)$ для $((p \to q) \wedge (q \to p))$, можно записывать утверждения об эквивалентности формул в виде тавтологий. Вот несколько таких эквивалентностей:

Теорема 10. *Формулы*

$$
\begin{aligned}
(p \wedge q) &\leftrightarrow (q \wedge p);\\
((p \wedge q) \wedge r) &\leftrightarrow (p \wedge (q \wedge r));\\
(p \vee q) &\leftrightarrow (q \vee p);\\
((p \vee q) \vee r) &\leftrightarrow (p \vee (q \vee r));\\
(p \wedge (q \vee r)) &\leftrightarrow ((p \wedge q) \vee (p \wedge r));\\
(p \vee (q \wedge r)) &\leftrightarrow ((p \vee q) \wedge (p \vee r));\\
\neg(p \wedge q) &\leftrightarrow (\neg p \vee \neg q);\\
\neg(p \vee q) &\leftrightarrow (\neg p \wedge \neg q);\\
(p \vee (p \wedge q)) &\leftrightarrow p;\\
(p \wedge (p \vee q)) &\leftrightarrow p;\\
(p \to q) &\leftrightarrow (\neg q \to \neg p);\\
p &\leftrightarrow \neg\neg p
\end{aligned}
$$

являются тавтологиями.

Доказательство. Первые четыре эквивалентности выражают коммутативность и ассоциативность конъюнкции и дизъюнкции. Проверим, например, вторую: левая и правая части истинны в единственном случае (когда все переменные истинны), и потому эквивалентны. (Для дизъюнкции удобнее смотреть, когда она ложна.)

Две следующие эквивалентности утверждают дистрибутивность — заметим, что в отличие от сложения и умножения в кольцах здесь верны оба свойства дистрибутивности. Проверить эквивалентность легко, если отдельно рассмотреть случаи истинного и ложного p.

Следующие два свойства, *законы Де Моргана*, легко проверить, зная, что конъюнкция истинна, а дизъюнкция ложна лишь в одном случае. Эти свойства иногда выражают словами: "конъюнкция двойственна дизъюнкции".

Далее следуют два очевидных *закона поглощения* (один из них мы уже упоминали).

За ними идёт правило *контрапозиции*, которое говорит, в частности, что утверждения "если x совершенно, то x чётно" и "если x нечётно, то x несовершенно" равносильны. Хотя оно и очевидно проверяется с помощью таблиц истинности, с ним связаны любопытные парадоксы. Вот один из них. С точки зрения формальной логики утверждения "кто не с нами, тот против нас" и "кто не против нас, тот с нами" равносильны.

Последнее (и очевидное) правило $p \leftrightarrow \neg\neg p$ называется *снятием двойного отрицания*. □

Отступление о пользе скобок. На самом деле наше определение истинности содержит серьёзный пробел. Чтобы обнаружить его, зададим себе вопрос: зачем нужны скобки в формулах? Представим себе, что мы изменим определение формулы, и будем говорить, что $P \wedge Q$ и $P \vee Q$ являются формулами для любых P и Q. Останутся ли наши рассуждения в силе?

Легко понять, что мы столкнёмся с трудностью при определении булевой функции, соответствующей формуле. В этом определении мы подставляли нули и единицы на место переменных и затем вычисляли значение формулы с помощью таблиц истинности для связок. Но теперь, когда мы изменили определение формулы, формула $p \wedge q \vee r$ может быть получена двумя способами — из формул $p \wedge q$ и r с помощью операции $\vee$ и из формул p и $q \vee r$ с помощью операции $\wedge$. Эти два толкования дадут разный результат при попытке вычислить значение $0 \wedge 0 \vee 1$.

Из сказанного ясно, что скобки нужны, чтобы гарантировать однозначность синтаксического разбора формулы. Точнее говоря, верно такое утверждение:

Теорема 11 (однозначность разбора). *Пропозициональная формула, не являющаяся переменной, может быть представлена ровно в одном из четырёх видов* $(A \wedge B)$, $(A \vee B)$, $(A \to B)$ *или* $\neg A$, *где* A *и* B *— некоторые формулы, причём* A *и* B *(в первых трёх случаях) восстанавливаются однозначно.*

Доказательство. Формальное доказательство можно провести так: назовём *скобочным итогом* разницу между числом открывающихся и закрывающихся скобок. Индукцией по построению формулы легко доказать такую лемму:

Лемма. Скобочный итог формулы равен нулю. Скобочный итог любого начала формулы неотрицателен и равен нулю, лишь если это начало совпадает со всей формулой, пусто или состоит из одних символов отрицания.

Слова "индукцией по построению" означают, что мы проверяем утверждение для переменных, а также доказываем, что если оно верно для формул A и B, то оно верно и для формул $(A \wedge B)$, $(A \vee B)$, $(A \to B)$ и $\neg A$.

После того как лемма доказана, разбор формулы проводится так: если она начинается с отрицания, то может быть образована лишь по четвертому правилу. Если же она начинается со скобки, то надо скобку удалить, а потом искать непустое начало, имеющее нулевой скобочный итог и не оканчивающееся на знак логической операции. Такое начало единственно (как легко проверить, используя лемму). Это начало и будет первой частью формулы. Тем самым формула разбирается однозначно. □

Нет смысла вдаваться в подробности этого (несложного) рассуждения: вообще-то алгоритмы разбора формул — это отдельная большая и практически важная тема (в первую очередь в связи с компиляторами). Приведённый нами алгоритм далеко не оптимален.

В дальнейшем мы будем опускать скобки, если они либо не играют роли (например, можно написать конъюнкцию трёх членов, не указывая порядок действий в силу ассоциативности), либо ясны из контекста.

§2. Полные системы связок

Рассматриваемая нами система пропозициональных связок $(\wedge, \vee, \to, \neg)$ *полна* в следующем смысле:

Теорема 12 (Полнота системы связок). *Любая булева функция n аргументов может быть записана в виде пропозициональной формулы.*

Доказательство. Проще всего пояснить это на примере. Пусть, например, булева функция $\varphi(p, q, r)$ задана таблицей 4.

p	q	r	$\varphi(p,q,r)$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$$\begin{aligned}(\neg p \wedge \neg q \wedge \neg r) \vee \\ \vee (\neg p \wedge q \wedge r) \vee \\ \vee (p \wedge q \wedge r)\end{aligned}$$

Таблица 4. Булева функция и задающая её формула.

В таблице есть три строки с единицами в правой колонке — три случая, когда булева функция истинна (равна 1). Напишем три конъюнкции, каждая из которых покрывает один случай (а в остальных строках ложна), и соединим их дизъюнкцией. Нужная формула построена.

Ясно, что аналогичная конструкция применима для любой таблицы (с любым числом переменных). □

Для формул подобного вида есть специальное название: формулы в *дизъюнктивной нормальной форме*. Более подробно: *литералом* называется переменная или отрицание переменной, *конъюнктом* называется произвольная конъюнкция литералов, а дизъюнктивной нормальной формой называется дизъюнкция конъюнктов. В нашем случае в каждый конъюнкт входит n литералов (где n — число переменных), а число конъюнктов равно числу строк с единицами и может меняться от нуля (тогда, правда, получается не совсем формула, а "пустая дизъюнкция и её можно заменить какой-нибудь всегда ложной формулой типа $p \wedge \neg p$) до 2^n (если булева функция всегда истинна).

Иногда полезна *конъюнктивная нормальная форма*, которая представляет собой конъюнкцию *дизъюнктов*. Каждый дизъюнкт состоит из литералов, соединённых дизъюнкциями. Теорему 12 можно теперь усилить так:

Теорема 13. *Всякая булева функция может быть выражена формулой, находящейся в дизъюнктивной нормальной форме, а также формулой, находящейся в конъюнктивной нормальной форме.*

Доказательство. Первая часть утверждения уже доказана. Вторая часть аналогична первой, надо только для каждой строки с нулём написать подходящий дизъюнкт.

Можно также представить функцию $\neg\varphi$ в дизъюнктивной нормальной форме, а затем воспользоваться законами Де Моргана, чтобы внести отрицание внутрь. □

Вообще говоря, определение нормальной формы не требует, чтобы в каждом конъюнкте (или дизъюнкте) встречались все переменные. (Повторять переменную больше одного раза смысла нет; если, например, переменная и её отрицание входят в одну конъюнкцию, то эта конъюнкция всегда ложна и её можно выбросить.)

Заметим, что при доказательстве теоремы 12 мы обошлись без имплликации. Это и не удивительно, так как она выражается через дизъюнкцию и

отрицание:

$$(p \to q) \;\leftrightarrow\; (\neg p \vee q)$$

(проверьте!). Мы могли бы обойтись только конъюнкцией и отрицанием, так как

$$(p \vee q) \;\leftrightarrow\; \neg(\neg p \wedge \neg q),$$

или только дизъюнкцией и отрицанием, так как

$$(p \wedge q) \;\leftrightarrow\; \neg(\neg p \vee \neg q)$$

(обе эквивалентности вытекают из законов Де Моргана; их легко проверить и непосредственно). Как говорят, система связок $\wedge, \neg$, а также система связок $\vee, \neg$ являются *полными*. (По определению это означает, что с их помощью можно записать любую булеву функцию.)

Задача 3. *Докажите, что система связок* $\neg, \to$ *полна. (Указание: как записать через них дизъюнкцию?)*

А вот без отрицания обойтись нельзя. Система связок $\wedge, \vee, \to$ неполна — и по очень простой причине: если все переменные истинны, то любая их комбинация, содержащая только указанные связки, истинна. (Как говорят, все эти связки "сохраняют единицу".)

В принципе мы не обязаны ограничиваться четырьмя рассмотренными связками. Любая булева функция может играть роль связки. Например, можно рассмотреть связку (p `notand` q), задаваемую эквивалентностью

$$(p \texttt{ notand } q) \;\leftrightarrow\; \neg(p \wedge q)$$

(словами: (p `notand` q) ложно, лишь если p и q истинны). Через неё выражается отрицание (p `notand` p), после чего можно выразить конъюнкцию, а затем, как мы знаем, и вообще любую функцию. (Знакомые с цифровыми логическими схемами малого уровня интеграции хорошо знакомы с этим утверждением: достаточно большой запас схем И-НЕ позволяет реализовать любую требуемую зависимость выхода от входов.)

Другая интересная полная система связок — сложение по модулю 2, конъюнкция и константа 1 (которую можно считать 0-арной связкой, задающей функцию от нуля аргументов). Представленные в этой системе булевы функции становятся полиномами с коэффициентами в кольце вычетов по модулю 2. Идея рассматривать булевы функции как полиномы (оказавшаяся неожиданно плодотворной в последние годы) была высказана в 1927 г. российским математиком Иваном Ивановичем Жегалкиным.

Назовём *мономом* конъюнкцию любого набора переменных или константу 1 (которую естественно рассматривать как конъюнкцию нуля переменных). Название это естественно, так как при наших соглашениях (1 обозначает истину, 0 — ложь) конъюнкция соответствует умножению.

Назовём *полиномом* сумму таких мономов по модулю 2 (это значит, что $0 \oplus 0 = 0$, $0 \oplus 1 = 1 \oplus 0 = 1$ и $1 \oplus 1 = 0$). Ясно, что два повторяющихся монома можно сократить (ведь сложение по модулю 2), так что будем рассматривать только полиномы без повторяющихся мономов. При этом, естественно, порядок членов в мономе (как и порядок мономов в полиноме) роли не играет, их можно переставлять.

Теорема 14 (о полиномах Жегалкина). *Всякая булева функция однозначно представляется таким полиномом.*

Доказательство. Существование искомого полинома следует из теоремы 13, так как конъюнкция есть умножение, отрицание — прибавление единицы, а дизъюнкцию можно через них выразить (получится $p + q + pq$). Надо только заметить, что степени не нужны: переменные принимают значения 0 и 1, так что x^n можно заменить на x.

Далее можно заметить, что полиномов столько же, сколько булевых функций, а именно 2^{2^n}. В самом деле, булева функция может принимать любое из двух значений в каждой из 2^n точек булева куба $\mathbb{B}^n$, а многочлен может включать или не включать любой из 2^n мономов. (Мономов ровно 2^n, потому что каждый моном включает или не включает любую из n переменных.) Поэтому избытка полиномов нет, и если любая функция представима полиномом, то единственным образом.

□

Если рассматривать произвольные булевы функции в качестве связок, возникает вопрос: в каком случае набор булевых функций образует полный базис? (Это значит, что любая булева функция представляется в виде композиции функций из набора, —е записывается в виде формулы, где связками служат функции набора.) Подобные вопросы вызывали в своё время большой интерес и были хорошо изучены. Начальным этапом явилось такое утверждение:

Теорема 15 (критерий Поста). *Набор булевых функций является полным тогда и только тогда, когда он не содержится целиком ни в одном из пяти следующих "предполных классов":*

- *монотонные функции;*
- *функции, сохраняющие нуль;*
- *функции, сохраняющие единицу;*

- *линейные функции;*
- *самодвойственные функции.*

(Функция f монотонна, *если она монотонно неубывает по каждому из своих аргументов. Функция* f сохраняет нуль/единицу, *если* $f(0,\dots,0)=0$ *(соответственно* $f(1,\dots,1)=1$*). Функция* f линейна, *если она представима многочленом, в котором все мономы содержат не более одной переменной. Наконец, функция* f *называется* самодвойственной, *если* $f(1-p_1,\dots,1-p_n)=1-f(p_1,\dots,p_n)$*.)*

Доказательство. Доказательство этого утверждения неложно, и мы его опускаем. □

§3. Исчисление высказываний

Напомним, что тавтологией мы называли пропозициональную формулу, истинную при всех значениях переменных. Оказывается, что все тавтологии можно получить из некоторого набора "аксиом"с помощью "правил вывода которые имеют чисто синтаксический характер и никак не апеллируют к смыслу формулы, её истинности и т.д. Эту задачу решает так называемое *исчисление высказываний (ИВ)*. В этой главе мы перечислим аксиомы и правила вывода этого исчисления, и приведём несколько доказательств *теоремы о полноте* (которая утверждает, что всякая тавтология выводима в исчислении высказываний).

Каковы бы ни были формулы A,B,C, следующие формулы называют *аксиомами исчисления высказываний*:

(1) $A\to(B\to A)$;
(2) $(A\to(B\to C))\to((A\to B)\to(A\to C))$;
(3) $(A\wedge B)\to A$;
(4) $(A\wedge B)\to B$;
(5) $A\to(B\to(A\wedge B))$;
(6) $A\to(A\vee B)$;
(7) $B\to(A\vee B)$;
(8) $(A\to C)\to((B\to C)\to(A\vee B\to C))$;
(9) $\neg A\to(A\to B)$;
(10) $(A\to B)\to((A\to\neg B)\to\neg A)$;
(11) $A\vee\neg A$.

Как говорят, мы имеем здесь одиннадцать "схем аксиом"; из каждой схемы можно получить различные конкретные аксиомы, заменяя входящие в неё буквы на пропозициональные формулы.

Единственным правилом вывода исчисления высказываний является правило со средневековым названием "modus ponens"(MP). Это правило разрешает получить (вывести) из формул A и $(A \to B)$ формулу B.

Выводом в исчислении высказываний называется конечная последовательность формул, каждая из которых есть аксиома или получается из предыдущих по правилу modus ponens.

Вот пример вывода (в нём первая формула является частным случаем схемы (1), вторая — схемы (2), а последняя получается из двух предыдущих по правилу modus ponens):

$$(p \to (q \to p)),$$
$$(p \to (q \to p)) \to ((p \to q) \to (p \to p)),$$
$$((p \to q) \to (p \to p)).$$

Пропозициональная формула A называется *выводимой* в исчислении высказываний, или *теоремой* исчисления высказываний, если существует вывод, в котором последняя формула равна A. Такой вывод называют выводом формулы A. (В принципе можно было бы и не требовать, чтобы формула A была последней — все дальнейшие формулы можно просто вычеркнуть.)

Как мы уже говорили, в исчислении высказываний выводятся все тавтологии и только они. Обычно это утверждение разбивают на две части: простую и сложную. Начнём с простой:

Теорема 16 (о корректности ИВ). *Всякая теорема исчисления высказываний есть тавтология.*

Доказательство. Несложно проверить, что все аксиомы — тавтологии. Для примера проделаем это для самой длинной аксиомы (точнее, схемы аксиом) — для второй. В каком случае формула

$$(A \to (B \to C)) \to ((A \to B) \to (A \to C))$$

(где A, B, C — некоторые формулы) могла бы быть ложной? Для этого посылка $A \to (B \to C)$ должна быть истинной, а заключение $(A \to B) \to (A \to C)$ — ложным. Чтобы заключение было ложным, формула $A \to B$ должна быть истинной, а формула $A \to C$ — ложной. Последнее означает, что A истинна, а C ложна. Таким образом, мы знаем, что A, $(A \to B)$ и $(A \to (B \to C))$ истинны. Отсюда следует, что B и $(B \to C)$ истинны, и потому C истинна — противоречие. Значит, наша формула не бывает ложной.

Корректность правила MP также очевидна: если формулы $(A \to B)$ и A всегда истинны, то по определению импликации формула B также всегда истинна. Таким образом, все формулы, входящие в выводы (все теоремы) являются тавтологиями. □

Гораздо сложнее доказать обратное утверждение.

Теорема 17 (о полноте ИВ). *Всякая тавтология есть теорема исчисления высказываний.*

Доказательство будет чуть позже. Прежде всего мы должны приобрести некоторый опыт построения выводов и использования аксиом.

Лемма 1. Какова бы ни была формула D, формула $(D \to D)$ является теоремой.

Докажем лемму, предъявив вывод формулы $(D \to D)$ в исчислении высказываний.

1. $(D \to ((D \to D) \to D)) \to ((D \to (D \to D)) \to (D \to D))$
   [аксиома 2 при $A = D$, $B = (D \to D)$, $C = D$];
2. $D \to ((D \to D) \to D)$ [аксиома 1];
3. $(D \to (D \to D)) \to (D \to D)$ [из 1 и 2 по правилу MP];
4. $D \to (D \to D)$ [аксиома 1];
5. $(D \to D)$ [из 3 и 4 по правилу MP].

Как видно, вывод даже такой простой тавтологии, как $(D \to D)$, требует некоторой изобретательности. Мы облегчим себе жизнь, доказав некоторое общее утверждение о выводимости.

Часто мы рассуждаем так: предполагаем, что выполнено какое-то утверждение A, и выводим различные следствия. После того как другое утверждение B доказано, мы вспоминаем, что использовали предположение A, и заключаем, что мы доказали утверждение $A \to B$. Следующая лемма, называемая иногда "леммой о дедукции показывает, что этот подход правомерен и для исчисления высказываний.

Пусть Γ — некоторое множество формул. *Выводом из* Γ называется конечная последовательность формул, каждая из которых является аксиомой, принадлежит Γ или получается из предыдущих по правилу MP. (Другими словами, мы как бы добавляем формулы из Γ к аксиомам исчисления высказываний — именно как формулы, а не как схемы аксиом.) Формула A *выводима из* Γ, если существует вывод из Γ, в котором она является последней формулой. В этом случае мы пишем $\Gamma \vdash A$. Если Γ пусто, то речь идёт о выводимости в исчислении высказываний, и вместо $\varnothing \vdash A$ пишут просто $\vdash A$.

Лемма 2 (о дедукции). Пусть Γ — множество формул. Тогда $\Gamma \vdash A \to B$ тогда и только тогда, когда $\Gamma \cup \{A\} \vdash B$.

Доказательство леммы имеет технический характер и мы его опустим.

Легче доказывается вторая форма теоремы о полноте Это доказательство теоремы о полноте обобщается на более сложные случаи (исчисление

предикатов, интуиционистское исчисление высказываний). Начнём с такого определения: множество формул Γ называется *совместным*, если существует набор значений переменных, при которых все формулы из Γ истинны. Заметим, что формула φ является тавтологией тогда и только тогда, когда множество, состоящее из единственной формулы $\neg\varphi$, не является совместным. Для случая одной формулы есть специальный термин: формула τ *выполнима*, если существуют значения переменных, при которых она истинна, то есть если множество $\{\tau\}$ совместно. Тавтологии — это формулы, отрицания которых не выполнимы.

Множество формул Γ называется *противоречивым*, если из него одновременно выводятся формулы A и $\neg A$. В этом случае из него выводятся вообще все формулы. (В противном случае Γ называется *непротиворечивым*.)

Теорема 18 (корректность исчисления высказываний, вторая форма). *Всякое совместное множество формул непротиворечиво.*

Доказательство. В самом деле, пусть совместное множество Γ противоречиво. Так как оно совместно, существуют значения переменных, при которых все формулы из Γ истинны. С другой стороны, из Γ выводится некоторая формула B и её отрицание. Может ли так быть?

Оказывается, что нет. Мы уже видели, что всякая выводимая формула истинна при всех значениях переменных (является тавтологией). Справедливо и несколько более общее утверждение: если $\Gamma \vdash A$ и при некоторых значениях переменных все формулы из Γ истинны, то и формула A истинна при этих значениях переменных. (Как и раньше, это легко доказывается индукцией по построению вывода A из Γ.)

В нашей ситуации это приводит к тому, что на выполняющем наборе значений переменных для Γ должны быть истинны обе формулы B и $\neg B$, что, разумеется, невозможно. □

Мы называем это утверждение другой формой теоремы о корректности исчисления высказываний, поскольку из него формально можно вывести, что всякая теорема является тавтологией: если A — теорема, то множество $\{\neg A\}$ противоречиво (из него выводятся A и $\neg A$), потому несовместно, значит, $\neg A$ всегда ложна, поэтому A всегда истинна.

Теорема 19 (полнота исчисления высказываний, вторая форма). *Любое непротиворечивое множество совместно.*

Доказательство можно найти в более полных учебниках, например в [2].

Итак, всякое непротиворечивое множество формул совместно. Отсюда легко следует, что всякая тавтология является теоремой. В самом деле, если

φ — тавтология, то множество $\{\neg\varphi\}$ несовместно, поэтому из $\neg\varphi$ выводится противоречие, поэтому $\vdash \neg\neg\varphi$, и по закону снятия двойного отрицания $\vdash \varphi$.

Кроме того, теорема о полноте во второй формулировке имеет такое очевидное следствие:

Теорема 20 (теорема компактности для исчисления высказываний). *Пусть* Γ *— множество формул, всякое конечное подмножество которого совместно. Тогда и всё множество* Γ *совместно.*

Доказательство. Как мы знаем, несовместность равносильна противоречивости, а вывод противоречия по определению может использовать лишь конечное число формул. □

§4. Предикаты: формулы и интерпретации

Начнём с примера. Пусть M — некоторое непустое множество, а R — бинарное отношение на нём, то есть подмножество декартова произведения $M \times M$. Вместо $\langle x, y\rangle \in R$ мы будем писать $R(x, y)$. Рассмотрим формулу

$$\forall x\, \exists y\, R(x, y).$$

Эта формула выражает некоторое свойство бинарного отношения R (для любого элемента $x \in M$ найдётся элемент, находящийся с ним в отношении R) и может быть истинна или ложна. Например, если M есть множество натуральных чисел $\mathbb{N}$, а R — отношение "строго меньше" (другими словами, R есть множество всех пар $\langle x, y\rangle$, для которых $x < y$), то эта формула истинна. А для отношения "строго больше" (на том же множестве) эта формула ложна.

Вопрос о том, будет ли истинна формула

$$\exists y\, R(x, y)$$

для данного множества M и для данного бинарного отношения R на нём, не имеет смысла, пока не уточнено, каково значение переменной x. Например, если $M = \mathbb{N}$ и $R(x, y)$ есть $x > y$, то эта формула будет истинной при $x = 3$ и ложной при $x = 0$. Для данных M и R она задаёт некоторое свойство элемента x и тем самым определяет некоторое подмножество множества M.

Перейдём к формальным определениям. Пусть M — непустое множество. Множество M^k состоит из всех последовательностей $\langle m_1, \ldots, m_k\rangle$ длины k, составленных из элементов множества M. Назовём *k-местной функцией* на множестве M любое отображение M^k в M (определённое на всём M^k). Синонимы: "функция k аргументов "функция валентности k "функция местности

k" и даже "функция арности k" (последнее слово происходит от слов "унарная" для функций одного аргумента, "бинарная" (операция) для функций двух аргументов и "тернарная" для трёх аргументов).

Назовём *k-местным предикатом* на множестве M любое отображение M^k в множество $\mathbb{B} = \{\mathbf{И},\mathbf{Л}\}$. Такой предикат будет истинным на некоторых наборах $\langle m_1, \ldots, m_k\rangle$ множества M и ложным на остальных наборах. Поставив ему в соответствие множество тех наборов, где он истинен, мы получаем взаимно однозначное соответствие между k-местными предикатами на M и подмножествами множества M^k. Говоря о предикатах, также употребляют термины "валентность "число аргументов" и др.

Мы будем рассматривать также функции и предикаты валентности нуль. Множество M^0 одноэлементно (содержит единственную последовательность длины 0). Поэтому функции $M^0 \to M$ отождествляются с элементами множества M, а нульместных предикатов ровно два — истинный и ложный.

Естественно, что в формулы будут входить не сами функции и предикаты, а обозначения для них, которые называют *функциональными* и *предикатными символами*. В качестве символов можно использовать любые знаки. Важно лишь, что каждому символу приписана валентность, которая определяет, со сколькими аргументами он может встречаться в формуле. Произвольный набор предикатных и функциональных символов, для каждого из которых указано неотрицательное число, называемое *валентностью*, мы будем называть *сигнатурой*.

Остаётся определить три вещи: что такое формула данной сигнатуры, что такое интерпретация данной сигнатуры и когда формула является истинной (в данной интерпретации).

Фиксируем некоторый набор символов, называемых *индивидными переменными*. Они предназначены для обозначения элементов множества, на котором определены функции и предикаты; обычно в таком качестве используют латинские буквы x, y, z, u, v, w с индексами. В каждой формуле будет использоваться конечное число переменных, так что счётного набора переменных нам хватит. Мы предполагаем, что переменные отличны от всех функциональных и предикатных символов сигнатуры (иначе выйдет путаница).

Определим понятие *терма* данной сигнатуры. Термом называется последовательность переменных, запятых, скобок и символов сигнатуры, которую можно построить по следующим правилам:

- Индивидная переменная есть терм.
- Функциональный символ валентности 0 есть терм.

- Если $t_1, \ldots, t_k$ — термы, а f — функциональный символ валентности $k > 0$, то $f(t_1, \ldots, t_k)$ есть терм.

В принципе можно было не выделять функциональные символы валентности 0 (которые также называют *константами*) в отдельную группу, но тогда бы после них пришлось писать скобки (как это делается в программах на языке Си).

Если A — предикатный символ валентности k, а $t_1, \ldots, t_k$ — термы, то выражение $A(t_1, \ldots, t_k)$ считается *атомарной формулой*. Кроме того, любой предикатный символ валентности 0 считается атомарной формулой.

Формулы строятся по таким правилам:

- Атомарная формула есть формула.
- Если φ — формула, то $\neg\varphi$ — формула.
- Если φ и ψ — формулы, то выражения $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \to \psi)$ также являются формулами.
- Если φ есть формула, а ξ — индивидная переменная, то выражения $\forall\xi\,\varphi$ и $\exists\xi\,\varphi$ являются формулами.

Во многих случаях в сигнатуру входит двуместный предикатный символ $=$, называемый *равенством*. По традиции вместо $=(t_1, t_2)$ пишут $(t_1 = t_2)$.

Итак, понятие формулы в данной сигнатуре полностью определено. Иногда такие формулы называют *формулами первого порядка* данной сигнатуры, или формулами *языка первого порядка* с данной сигнатурой.

Наш следующий шаг — определение *интерпретации* данной сигнатуры. Пусть фиксирована некоторая сигнатура σ. Чтобы задать интерпретацию сигнатуры σ, необходимо:

- указать некоторое непустое множество M, называемое *носителем* интерпретации;
- для каждого предикатного символа сигнатуры σ указать предикат с соответствующим числом аргументов, определённый на множестве M (как мы уже говорили, 0-местным предикатным символам ставится в соответствие либо **И**, либо **Л**);
- для каждого функционального символа сигнатуры σ указать функцию соответствующего числа аргументов с аргументами и значениями из M (в частности, для 0-местных функциональных символов надо указать элемент множества M, с ними сопоставляемый).

Если сигнатура включает в себя символ равенства, то среди её интерпретаций выделяют *нормальные* интерпретации, в которых символ равенства интерпретируется как совпадение элементов множества M.

Приведём несколько примеров сигнатур, используемых в различных теориях.

Сигнатура теории упорядоченных множеств включает в себя два двуместных предикатных символа (равенство и порядок) и не имеет функциональных символов. Здесь также вместо $\leq (x, y)$ по традиции пишут $x \leq y$.

Аксиомы порядка (рефлексивность, антисимметричность, транзитивность) могут быть записаны формулами этой сигнатуры. Например, требование антисимметричности записывается так:

$$\forall x\, \forall y (((x \leq y) \wedge (y \leq x)) \to (x = y)).$$

Иногда в сигнатуру теории упорядоченных множеств вместо символа $\leq$ включают символ $<$; большой разницы тут нет.

Сигнатуру теории групп можно выбирать по-разному. Можно считать, что (помимо равенства) она имеет двуместный функциональный символ $\times$ (который по традиции записывают между множителями), константу (нульместный функциональный символ) 1 и одноместный функциональный символ $\mathrm{inv}(x)$ для обращения. Тогда аксиомы теории групп записываются с использованием лишь кванторов всеобщности:

$$\forall x\, \forall y\, \forall z (((x \times y) \times z) = (x \times (y \times z))),$$
$$\forall x\, (((x \times 1) = x) \wedge ((1 \times x) = x)),$$
$$\forall x\, (((x \times \mathrm{inv}(x)) = 1) \wedge ((\mathrm{inv}(x) \times x) = 1)).$$

Если не включать операцию обращения в сигнатуру, придётся использовать квантор существования и переписать последнюю аксиому так:

$$\forall x\, \exists y\, (((x \times y) = 1) \wedge ((y \times x) = 1)).$$

Сигнатура теории множеств содержит два двуместных предикатных символа: для принадлежности и для равенства. Аксиомы теории множеств можно записывать в виде формул этой сигнатуры.

Докажите основные эквивалентности, содержащие кванторы.

(1) $\overline{\forall x\ P(x)} \equiv \exists x\ \overline{P(x)}$
(2) $\overline{\exists x\ P(x)} \equiv \forall x\ \overline{P(x)}$
(3) $\forall x \forall y\ P(x, y) \equiv \forall y \forall x\ P(x, y)$
(4) $\exists x \exists y\ P(x, y) \equiv \exists y \exists x\ P(x, y)$
(5) $\forall x\ (P(x) \wedge Q(x)) \equiv \forall x\ P(x) \wedge \forall x\ Q(x))$
(6) $\exists x\ (P(x) \vee Q(x)) \equiv \exists x\ P(x) \vee \exists x\ Q(x))$
(7) $\forall x\ (P(x) \vee Q(y)) \equiv \forall x\ P(x) \vee Q(y)$
(8) $\exists x\ (P(x) \wedge Q(y)) \equiv \exists x\ P(x) \wedge Q(y)$
(9) $\exists y \forall x\ P(x, y) \Rightarrow \forall x \exists y\ P(x, y) \equiv 1$

§5. Определение истинности

Из приведённых выше примеров, вероятно, понятен смысл формулы, то есть ясно, в каких интерпретациях данной сигнатуры и для каких элементов формула истинна. Тем не менее для любителей строгости мы приведём формальное определение истинности. (Его детали понадобятся, когда мы будем проверять истинность выводимых формул, см. раздел 11.)

Прежде всего, определим формально понятие *параметра* формулы (переменной, от значения которой может зависеть истинность формулы). Согласно этому определению, скажем, формула $\forall x\,\exists y\,A(x,y)$ не имеет параметров, а формулы $\exists y\,A(x,y)$ и $(A(x)\wedge\forall x\,B(x,x))$ имеют единственный параметр x. Вот как выглядит это определение:

- Параметрами терма являются все входящие в него индивидные переменные.
- Параметрами атомарной формулы являются параметры всех входящих в неё термов.
- Параметры формулы $\neg\varphi$ те же, что у формулы φ.
- Параметрами формул $(\varphi\wedge\psi)$, $(\varphi\vee\psi)$ и $(\varphi\to\psi)$ являются все параметры формулы φ, а также все параметры формулы ψ.
- Параметрами формул $\forall\xi\,\varphi$ и $\exists\xi\,\varphi$ являются все параметры формулы φ, кроме переменной ξ.

Параметры иногда называют *свободными переменными* формулы. Заметим, что формула может иметь одновременно параметр x и использовать (в другом месте) квантор $\forall x$. Как говорят в этом случае, одна и та же переменная имеет *свободные* и *связанные* вхождения. Свободное вхождение переменной — это такое вхождение, которое не входит в область действия одноимённого квантора. Если аккуратно определить эту область действия, несложно проверить, что параметры формулы — это как раз переменные, имеющие свободные вхождения.

Теперь мы хотим определить понятие формулы, истинной в данной интерпретации при данных значениях параметров. Технически проще считать, что всем индивидным переменным приписаны какие-то значения, а потом доказать, что переменные, не являющиеся параметрами, не влияют на истинность формулы.

Итак, пусть фиксирована сигнатура и некоторая интерпретация этой сигнатуры. *Оценкой* назовём отображение, которое ставит в соответствие каждой индивидной переменной некоторый элемент множества, являющегося носителем интерпретации. Этот элемент будем называть *значением переменной* при данной оценке.

Определим индуктивно *значение терма* t при данной оценке π, которое мы будем обозначать $[t](\pi)$.

- Для переменных оно уже определено.
- Если t является константой (нульместным функциональным символом), то $[t](\pi)$ не зависит от π и равно значению этой константы при данной интерпретации (напомним, в интерпретации с каждой константой сопоставляется некоторый элемент носителя).
- Если t имеет вид $f(t_1, \ldots, t_m)$, где f — функциональный символ валентности m, а $t_1, \ldots, t_m$ — термы, то $[t](\pi)$ есть $[f]([t_1](\pi), \ldots, [t_m](\pi))$, где $[f]$ есть функция, соответствующая символу f в нашей интерпретации, а $[t_i](\pi)$ есть значение терма t_i при оценке π.

Теперь можно определить *значение формулы* φ при данной оценке π в данной интерпретации, которое обозначается $[\varphi](\pi)$ и может быть равно **И** или **Л**; в первом случае формула называется *истинной*, во втором — *ложной*. Это определение также индуктивно:

- Значение атомарной формулы $A(t_1, \ldots, t_m)$ определяется как
$$[A]([t_1](\pi), \ldots, [t_m](\pi)),$$
где $[A]$ — предикат, соответствующий предикатному символу A в рассматриваемой интерпретации. Если формула представляет собой нульместный предикатный символ, то её значение не зависит от оценки и есть значение этого символа.
- $[\neg\varphi](\pi)$ определяется как $\neg[\varphi(\pi)]$, где $\neg$ понимается как операция в $\mathbb{B}$. Другими словами, формула $\neg\varphi$ истинна при оценке π тогда и только тогда, когда формула φ ложна при этой оценке.
- $[\varphi \wedge \psi](\pi)$ определяется как $[\varphi](\pi) \wedge [\psi](\pi)$, где $\wedge$ в правой части понимается как операция в $\mathbb{B}$. (Другими словами, формула $(\varphi \wedge \psi)$ истинна при оценке π тогда и только тогда, когда обе формулы φ и ψ истинны при этой оценке.) Аналогичным образом $[\varphi \vee \psi](\pi)$ определяется как $[\varphi](\pi) \vee [\psi](\pi)$, а $[\varphi \to \psi](\pi)$ — как $[\varphi](\pi) \to [\psi](\pi)$.
- Формула $\forall\xi\,\varphi$ истинна на оценке π тогда и только тогда, когда формула φ истинна на любой оценке π', которая совпадает с π всюду, кроме значения переменной ξ (которое в оценке π' может быть любым). Другими словами, если обозначить через $\pi + (\xi \mapsto m)$ оценку, при которой значение переменной ξ равно m, а остальные переменные принимают те же значения, что и в оценке π, то
$$[\forall\xi\,\varphi](\pi) = \bigwedge_{m \in M} [\varphi](\pi + (\xi \mapsto m)).$$

(В правой части стоит бесконечная конъюнкция, которая истинна, если все её члены истинны.)

- Формула $\exists\xi\,\varphi$ истинна на оценке π тогда и только тогда, когда формула φ истинна на некоторой оценке π', которая совпадает с π всюду, кроме значения переменной ξ (которое в оценке π' может быть любым). Другими словами,

$$[\forall\xi\,\varphi](\pi) = \bigvee_{m\in M} [\varphi](\pi + (\xi \mapsto m)).$$

(В правой части стоит бесконечная дизъюнкция, которая истинна, если хотя бы один из её членов истинен.)

Заметим, что в двух последних пунктах значение переменной ξ в оценке π не играет роли. Это позволяет легко доказать (индукцией по построению формулы) такое утверждение: если две оценки π_1 и π_2 придают одинаковые значения всем параметрам формулы φ, то $[\varphi](\pi_1) = [\varphi](\pi_2)$. Другими словами, истинность формулы определяется значениями её параметров.

Формула называется *замкнутой*, если она не имеет параметров. Замкнутые формулы называют также *суждениями*. Как мы доказали, истинность замкнутой формулы определяется выбором интерпретации (и не зависит от значений переменных).

§6. Выразимые предикаты

Пусть фиксирована некоторая сигнатура σ и её интерпретация с носителем M. Мы хотим определить понятие *выразимого* (с помощью формулы данной сигнатуры в данной интерпретации) k-местного предиката.

Выберем k переменных $x_1, \ldots, x_k$. Рассмотрим произвольную формулу φ, все параметры которой содержатся в списке $x_1, \ldots, x_k$. Истинность этой формулы зависит только от значений переменных $x_1, \ldots, x_k$. Тем самым возникает отображение $M^k \to \mathbb{B} = \{\mathbf{И},\mathbf{Л}\}$, то есть некоторый k- местный предикат на M. Говорят, что этот предикат *выражается* формулой φ. Все предикаты, которые можно получить таким способом, называются *выразимыми* (Ясно, что конкретный выбор списка переменных роли не играет.) Соответствующие им подмножества множества M^k (области истинности выразимых предикатов) также называют выразимыми.

Очевидно, что пересечение, объединение и разность двух выразимых множеств являются выразимыми.

Пример. Сигнатура содержит одноместный функциональный символ S и двуместный предикатный символ равенства ($=$). Рассмотрим интерпретацию этой сигнатуры. В качестве носителя выберем натуральный ряд $\mathbb{N}$. Символ S будет обозначать функцию прибавления единицы (можно считать S сокращением от слова successor — последователь). Знак равенства интерпретируется как совпадение элементов.

Легко проверить, что одноместный предикат "быть нулём" выразим в этой интерпретации, несмотря на то, что константы для нуля в сигнатуре не предусмотрено. В самом деле, он выражается формулой

$$\neg\exists y(x = S(y))$$

с единственным параметром x.

Ещё проще выразить в этой сигнатуре двуместный предикат "быть больше на 2 при этом даже не нужны кванторы: $y = S(S(x))$.

Можно доказать, что в этой сигнатуре кванторы почти не увеличивают набор выразимых предикатов: всякий выразимый предикат будет выражаться бескванторной формулой (возможно, гораздо более длинной), если добавить к сигнатуре константу 0.

§7. Выразимость в арифметике

Рассмотрим сигнатуру, имеющую два двуместных функциональных символа — сложение и умножение (как обычно, мы будем писать $x + y$ вместо $+(x, y)$ и т.д.) и двуместный предикатный символ равенства. Рассмотрим интерпретацию этой сигнатуры, носителем которой является множество $\mathbb{N}$ натуральных чисел, а сложение, умножение и равенство интерпретируются стандартным образом.

Выразимые с помощью формул этой сигнатуры предикаты называются *арифметическими* и играют в математической логике важную роль. Соответствующие множества также называются *арифметическими*. О них подробно рассказано в другой главе; оказывается, что почти всякое множество, которое можно описать словами, является арифметическим.

Для начала мы установим арифметичность довольно простых предикатов.

- Предикат $x \leq y$ является арифметическим. В самом деле, его можно записать как $\exists z\,(x + z = y)$.
- Предикаты $x = 0$ и $x = 1$ являются арифметическими. В самом деле, $x = 0$ тогда и только тогда, когда $x \leq y$ для любого y (а также когда

$x + x = x$). А $x = 1$ тогда и только тогда, когда x представляет собой наименьшее число, отличное от нуля. (Можно также воспользоваться тем, что $y \cdot 1 = y$ при любом y.)

- Вообще для любого фиксированного числа c предикат $x = c$ является арифметическим. (Например, можно написать сумму из большого числа единиц.)
- Полезно такое общее наблюдение: если мы уже установили, что какой-то предикат является арифметическим, то в дальнейшей его можно использовать в формулах, как если бы он входил в сигнатуру, поскольку его всегда можно заменить на выражающую его формулу.
- Предикат $x|y$ (число x является делителем числа y), очевидно, арифметичен (формула $\exists z\,(xz = y)$).
- Предикат "x — простое число" арифметичен. В самом деле, число просто, если оно отлично от 1 и любой его делитель равен 1 или самому числу. Это сразу же записывается в виде формулы.
- Операции частного и остатка при делении арифметичны (в том смысле, что трёхместные предикаты "q есть частное при делении a на b" и "r есть остаток при делении a на b" арифметичны. Например, первый из них записывается формулой $\exists r\,((a = bq + r) \wedge (r < b))$ (как мы уже говорили, использование арифметического предиката $(r < b)$ не создаёт проблем).
- Этот список можно продолжать: для многих предикатов их определение по существу уже является нужной формулой. Например, свойства "быть наибольшим общим делителем "быть наименьшим общим кратным "быть взаимно простыми" все относятся к этой категории.
- Предикат "быть степенью двойки" является арифметическим (хотя это и не столь очевидно, как в предыдущих примерах). В самом деле, это свойство можно переформулировать так: любой делитель либо равен единице, либо чётен.

Последнее из наших рассуждений годится для степеней тройки и вообще для степеней любого простого числа. Однако уже для степеней четвёрки оно не проходит, и, пожалуй, мы подошли к границе, где без некоторого общего метода не обойтись.

Два наиболее известных способа доказывать арифметичность основаны на возможности "кодирования" конечных множеств и последовательностей. Один восходит к Гёделю (так называемая β-функция Гёделя), второй изложен в книге "Теория формальных систем. Её написал Р. Смаллиан, известный как автор книги, которая имеет парадоксальное название "Как же называется эта книга?).

В некоторых отношениях метод Гёделя предпочтительней, и мы рассказываем о нём ниже, но сейчас для разнообразия рассмотрим другой способ. Зафиксируем взаимно однозначное соответствие между натуральными числами и двоичными словами:

0	1	2	3	4	5	6	7	8	...
Λ	0	1	00	01	10	11	000	001	...

Это соответствие задаётся так: чтобы получить слово, соответствующее числу n, надо записать $n+1$ в двоичной системе и удалить первую единицу. Например, нулю соответствует пустое слово Λ, числу 15 — слово 0000 и т.д. Теперь можно говорить об арифметичности предикатов, определённых на двоичных словах, имея в виду арифметичность соответствующих предикатов на $\mathbb{N}$.

- Предикат "слово x состоит из одних нулей" арифметичен. В самом деле, при переходе к числам ему соответствует предикат $x+1$ есть степень двойки который (как мы видели) арифметичен.
- Предикат "слова x и y имеют одинаковую длину" арифметичен. В самом деле, это означает, что найдётся степень двойки c, для которой $c-1 \leq x, y < 2c-1$ (именно такой промежуток заполняют числа, которым соответствуют слова одной длины).
- Предикат "слово z является конкатенацией слов x и y" (проще говоря, z получается приписыванием y справа к слову x) арифметичен. В самом деле, его можно выразить так: найдётся слово y' из одних нулей, имеющее ту же длину, что и слово y, при этом $(z+1) = (x+1)(y'+1) + (y-y')$ (умножение на $y'+1$ соответствует дописыванию нулей, а добавление $y-y'$ заменяет нули на буквы слова y).
- Предикат "слово x является началом слова y" арифметичен. В самом деле, это означает, что существует слово t, при котором y есть конкатенация x и t.
- То же самое верно для предикатов "x есть конец слова y "x есть подслово слова y" (последнее означает, что найдутся слова u и v, для которых y есть конкатенация u, x и v; конкатенация трёх слов выразима через конкатенацию двух).

Последнее утверждение не упоминает явно о словах, и больше они нам не понадобятся: достаточно знать, что конечные множества натуральных чисел можно кодировать парами натуральных чисел в описанном смысле.

Теперь мы можем выразить, что число x является степенью числа 4, следующим образом: существует конечное множество U, которое содержит число x и обладает таким свойством: всякий элемент $u \in U$ либо равен 1, либо делится на 4 и $u/4$ также принадлежит U. Теперь надо везде заменить множество U на его код u_1, u_2, а утверждение $x \in U$ на $S(x, u_1, u_2)$, где S — построенный нами кодирующий предикат.

Задача 4. *Покажите, что двуместный предикат "x есть n-ое по порядку простое число" арифметичен.*

§8. Невыразимые предикаты: автоморфизмы

Мы видели, как можно доказать выразимость некоторых свойств. Сейчас мы покажем, каким образом можно доказывать невыразимость.

Начнём с такого примера. Пусть сигнатура содержит двуместный предикат равенства ($=$) и двуместную операцию сложения ($+$). Рассмотрим её интерпретацию, носителем которой являются целые числа, а равенство и сложение интерпретируются стандартным образом. Оказывается, что предикат $x > y$ не является выразимым.

Причина очевидна: с точки зрения сложения целые числа устроены симметрично, положительные ничем не отличаются от отрицательных. Если мы изменим знак у всех переменных, входящих в формулу, то её истинность не может измениться. Но при этом $x > y$ заменится на $x < y$, и потому это свойство не является выразимым.

Формально говоря, надо доказывать по индукции такое свойство: если формула φ указанной сигнатуры истинна при оценке π, то она истинна и при оценке π', в которой значения всех переменных меняют знак. (Подробно мы объясним это в общей ситуации дальше.)

Сформулируем общую схему, которой следует это рассуждение. Пусть имеется некоторая сигнатура σ и интерпретация этой сигнатуры, носителем которой является множество M. Взаимно однозначное отображение $\alpha\colon M \to M$ называется *автоморфизмом* интерпретации, если все функции и предикаты, входящие в интерпретацию, устойчивы относительно α. При этом k-местный предикат P называется *устойчивым* относительно α, если

$$P(\alpha(m_1), \ldots, \alpha(m_k)) \Leftrightarrow P(m_1, \ldots, m_k)$$

для любых элементов $m_1, \ldots, m_k \in M$. Аналогичным образом k-местная функция f называется *устойчивой* относительно α, если

$$f(\alpha(m_1), \ldots, \alpha(m_k)) = \alpha(f(m_1, \ldots, m_k)).$$

Это определение обобщает стандартное определение автоморфизма для групп, колец, полей и т.д.

Теорема 21. *Предикат, выразимый в данной интерпретации, устойчив относительно её автоморфизмов.*

Доказательство. Проведём доказательство этого (достаточно очевидного) утверждения формально.

Пусть π — некоторая оценка, то есть отображение, ставящее в соответствие всем индивидным переменным некоторые элементы носителя. Через $\alpha \circ \pi$ обозначим оценку, которая получится, если к значению каждой переменной применить отображение α; другими словами, $\alpha \circ \pi(\xi) = \alpha(\pi(\xi))$ для любой переменной ξ.

Первый шаг состоит в том, чтобы индукцией по построению терма t доказать такое утверждение: значение терма t при оценке $\alpha \circ \pi$ получается применением α к значению терма t при оценке π:

$$[t](\alpha \circ \pi) = \alpha([t](\pi)).$$

Для переменных это очевидно, а шаг индукции использует устойчивость всех функций интерпретации относительно α.

Теперь индукцией по построению формулы φ легко доказать такое утверждение:

$$[\varphi](\alpha \circ \pi) = [\varphi](\pi).$$

Мы не будем выписывать эту проверку; скажем лишь, что взаимная однозначность α используется, когда мы разбираем случай кванторов. (В самом деле, если с одной стороны изоморфизма берётся какой-то объект, то взаимная однозначность позволяет взять соответствующий ему объект с другой стороны изоморфизма.) □

Теорема 21 позволяет доказать невыразимость какого-то предиката, предъявив автоморфизм интерпретации, относительно которого интересующий нас предикат неустойчив. Вот несколько примеров:

- $(\mathbb{Z}, =, <)$ Сигнатура содержит равенство и отношение порядка. Интерпретация: целые числа. Невыразимый предикат: $x = 0$. Автоморфизм: $x \mapsto x + 1$.
- $(\mathbb{Q}, =, <, +)$ Сигнатура содержит равенство, отношение порядка и операцию сложения. Интерпретация: рациональные числа. Невыразимый предикат: $x = 1$. Автоморфизм: $x \mapsto 2x$.

Заметим, что сложение позволяет выразить предикат $x = 0$. Кроме того, отметим, что вместо рациональных чисел можно взять действительные (но не целые, так как в этом случае единица описывается как наименьшее число, большее нуля).

- $(\mathbb{R}, =, <, 0, 1)$ Сигнатура содержит равенство, порядок и константы 0 и 1. Интерпретация: действительные числа. Невыразимый предикат: $x = 1/2$. (Автоморфизм упорядоченного множества $\mathbb{R}$, сохраняющий 0 и 1, но не $1/2$, построить легко.)
- $(\mathbb{R}, =, +, 0, 1)$ Сигнатура содержит равенство, сложение, константы 0 и 1. Интерпретация: действительные числа. Одноместный предикат $x = \gamma$ выразим для рациональных γ и невыразим для иррациональных γ.

  В самом деле, выразимость для рациональных γ очевидна. Невыразимость для иррациональных γ следует из того, что для любых двух иррациональных γ_1 и γ_2 существует автоморфизм, переводящий γ_1 в γ_2. (В самом деле, рассмотрим $\mathbb{R}$ как бесконечномерное векторное пространство над $\mathbb{Q}$. Векторы $1, \gamma_1$ линейно независимы и потому их можно дополнить до базиса Гамеля (подробности смотри в книжке по теории множеств [**1**]). Сделаем то же самое с векторами $1, \gamma_2$. Получатся равномощные базисы, после чего мы берём $\mathbb{Q}$-линейный оператор, переводящий 1 в 1 и γ_1 в γ_2.)
- $(\mathbb{C}, =, +, \times, 0, 1)$ В сигнатуру входят предикат равенства, операции сложения и умножения, а также константы 0 и 1. Интерпретация: комплексные числа. Предикат $x = \gamma$, где γ — некоторое комплексное число, выразим для рациональных γ и невыразим для иррациональных γ.

  В самом деле, если γ иррационально, то оно может быть алгебраическим или трансцендентным. В первом случае рассмотрим многочлен из $\mathbb{Q}[x]$ минимальной степени, обращающийся в 0 в точке γ; по предположению он имеет степень больше 1 и потому имеет другой корень γ'. В алгебре доказывается, что существует автоморфизм $\mathbb{C}$ над $\mathbb{Q}$, переводящий γ в γ'.

  В случае трансцендентного γ мы используем такой факт: для любых трансцендентных $\gamma_1, \gamma_2 \in \mathbb{C}$ существует автоморфизм поля $\mathbb{C}$ над $\mathbb{Q}$, который переводит γ_1 в γ_2.

  Отметим, что для поля $\mathbb{R}$ вместо $\mathbb{C}$ такое рассуждение не проходит, так как это поле не имеет нетривиальных автоморфизмов. (Отношение порядка выразимо: положительные числа суть квадраты, поэтому любой автоморфизм сохраняет порядок. Поскольку автоморфизм

оставляет на месте все рациональные числа, он должен быть тождественным.)

(В этом случае предикат $x = \gamma$ выразим тогда и только тогда, когда γ — алгебраическое число.)

Задача 5. *Покажите, что предикат $y = x + 1$ невыразим в интерпретации $(\mathbb{Z}, =, f)$, где f — одноместная функция $x \mapsto (x + 2)$.*

Задача 6. *Покажите, что предикат $x = 2$ невыразим в множестве целых положительных чисел с предикатами равенства и "x делит y".*

§9. Исчисление предикатов: общезначимые формулы

Исчисление высказываний позволяло выводить все тавтологии из некоторого набора базисных тавтологий (названных аксиомами) с помощью некоторых правил вывода (на самом деле единственного правила modus ponens). Сейчас мы хотим решить аналогичную задачу для формул первого порядка. Соответствующее исчисление называется *исчислением предикатов*.

Пусть фиксирована некоторая сигнатура σ. Формула φ этой сигнатуры (возможно, с параметрами) называется *общезначимой*, если она истинна в любой интерпретации сигнатуры σ на любой оценке.

Общезначимые формулы в логике предикатов играют ту же роль, что тавтологии в логике высказываний. Между ними есть и формальная связь: если взять любую тавтологию и вместо входящих в неё пропозициональных переменных подставить произвольные формулы сигнатуры σ, получится общезначимая формула. В самом деле, пусть есть некоторая интерпретация сигнатуры σ и некоторая оценка (то есть фиксированы значения индивидных переменных). Тогда каждая из подставленных формул станет истинной или ложной, а значение всей формулы определяется с помощью таблиц истинности для логических связок, то есть по тем же правилам, что в логике высказываний.

Конечно, бывают и другие общезначимые формулы, не являющиеся частным случаями пропозициональных тавтологий. Например, формула

$$\forall x\, A(x) \to \exists y A(y)$$

общезначима (здесь существенно, что носитель любой интерпретации непуст). Другие примеры общезначимых формул (во втором случае φ — произвольная формула):

$$\exists y\, \forall x\, B(x, y) \to \forall x\, \exists y\, B(x, y), \qquad \neg\forall x\, \neg\varphi \to \exists x\, \varphi.$$

Многие вопросы можно сформулировать как вопросы об общезначимости некоторых формул. Например, можно записать свойства рефлексивности, транзитивности и антисимметричности в виде формул R, T и A сигнатуры $(=, <)$ и затем написать формулу

$$R \wedge T \wedge A \to \exists x \, \forall y \, ((y < x) \vee (y = x)).$$

Общезначимость этой формулы означала бы, что любое линейно упорядоченное множество имеет наибольший элемент, так что она не общезначима.

Две формулы φ и ψ (с параметрами или без) называются *эквивалентными*, если в любой интерпретации и на любой оценке, на которой истинна одна из них, истинна и другая. Это определение равносильно такому: формула $\varphi \leftrightarrow \psi$ общезначима. Здесь, напомним, $\varphi \leftrightarrow \psi$ есть сокращение для $((\varphi \to \psi) \wedge (\psi \to \varphi))$.

Общезначимость любой формулы φ очевидно равносильна общезначимости её *замыкания* — формулы, которая получится, если слева к φ приписать кванторы всеобщности по всем параметрам.

Двойственное к общезначимости понятие — выполнимость. Формула называется *выполнимой*, если она истинна в некоторой интерпретации на некоторой оценке. Очевидно, формула φ общезначима тогда и только тогда, когда формула $\neg\varphi$ не является выполнимой.

Чтобы проверить, является ли формула тавтологией, достаточно подставить в неё все возможные наборы значений переменных. Хотя этот процесс может быть на практике невыполним (наборов слишком много), теоретически мы имеем простой алгоритм проверки, является ли формула тавтологией. Для общезначимых формул в общем случае такого алгоритма не существует (теорема Чёрча); он есть только для очень ограниченных классов формул. Например, если сигнатура содержит только нульместные предикатные символы, то задача по существу сводится к проверке тавтологичности (в этом случае кванторы фиктивны). Чуть более сложен случай с одноместными предикатами.

§10. Аксиомы и правила вывода

Возвратимся к нашей задаче: какие аксиомы и правила вывода нам нужны, чтобы получить все общезначимые формулы некоторой сигнатуры σ? Естественно использовать все схемы аксиом (1) - (11) исчисления высказываний (раздел 3), но только вместо букв A, B и C теперь можно подставлять произвольные формулы сигнатуры σ. Теорема о полноте исчисления высказываний гарантирует, что после этого мы сможем вывести любой частный случай любой пропозициональной тавтологии (то есть любую формулу, которая

получается из пропозициональной тавтологии заменой пропозициональных переменных на формулы сигнатуры σ). В самом деле, возьмём вывод этой тавтологии в исчислении высказываний (которое, как мы знаем, полно) и выполним соответствующую замену во всех формулах этого вывода.

Почти столь же просто понять, что ничего другого такие аксиомы не дадут: если пользоваться лишь схемами аксиом (1)-(11), разрешая брать в них в качестве A, B, C произвольные формулы сигнатуры σ, а в качестве правила вывода использовать modus ponens, то все выводимые формулы будут частными случаями пропозициональных тавтологий. В самом деле, если какая-то подформула начинается с квантора, то в выводе она может встречаться только как единое целое, то есть такая подформула ведёт себя как пропозициональная переменная.

Это наблюдение скорее тривиально, чем удивительно — если среди наших аксиом и правил вывода нет ничего о смысле кванторов, то формулы, начинающиеся с кванторов, будут вести себя как неделимые блоки. Таким образом, нам нужны аксиомы и правила вывода, отражающие интуитивный смысл кванторов.

Вспомним, как выглядели аксиомы исчисления высказываний. У нас было два типа аксиом для конъюнкции и дизъюнкции: одни говорили, что из них следует (например, из $A \wedge B$ следовало B), а другие — как их можно доказать (например, аксиома $(A \to (B \to (A \wedge B)))$ говорила, что для доказательства $(A \wedge B)$ надо доказать A и B). Кванторы всеобщности и существования в некотором смысле аналогичны конъюнкции и дизъюнкции, и аксиомы для них тоже будут похожими. Например, среди аксиом будет формула

$$\forall x\, A(x) \to A(t),$$

где A — одноместный предикатный символ нашей сигнатуры, а t — константа, переменная или вообще любой терм. (Если A верно для всех x, то оно должно быть верно и для нашего конкретного t. Можно сказать и так: из "бесконечной конъюнкции" всех $A(x)$ вытекает один из её членов.)

Конечно, такую аксиому надо иметь не только для одноместного предикатного символа A, но для любой формулы φ, любой переменной ξ и любого терма t. Естественно сказать, что если φ — любая формула, а t — любой терм, то формула

$$\forall \xi\, \varphi \to \varphi(t/\xi),$$

где $\varphi(t/\xi)$ обозначает результат подстановки t вместо всех вхождений переменной ξ в формулу φ, является аксиомой. (Запись $\varphi(t/\xi)$ можно читать как "фи от тэ вместо кси".)

К сожалению, всё не так просто. Например, если формула φ имеет вид

$$A(x) \wedge \exists x\, B(x, x),$$

то подстановка терма $f(y)$ вместо x даст абсурдное выражение

$$A(f(y)) \wedge \exists f(y)\, B(f(y), f(y)),$$

вообще не являющееся формулой. А если подставить $f(y)$ только внутри A и B, то получится выражение

$$A(f(y)) \wedge \exists x\, B(f(y), f(y)),$$

которое хотя и будет формулой, но имеет совсем не тот смысл, который нам нужен.

Конечно, в данном случае по смыслу ясно, что подставлять $f(y)$ надо лишь вместо самого первого вхождения переменной x. Но если мы хотим определить формальную систему аксиом и правил вывода, то надо дать формальные определения.

Для каждого квантора в формуле рассмотрим его *область действия* — начинающуюся с него подформулу. *Свободным вхождением* индивидной переменной в формулу называется вхождение, не попадающее в область действия одноимённого квантора. Легко понять, что это определение можно переформулировать индуктивно:

- любое вхождение переменной в терм свободно;
- свободные вхождения переменной в формулу φ являются её свободными вхождениями в формулу $\neg\varphi$;
- свободные вхождения любой переменной в одну из формул φ и ψ являются свободными вхождениями в $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$ и $(\varphi \to \psi)$;
- переменная ξ не имеет свободных вхождений в формулы $\forall\xi\,\varphi$ и $\exists\xi\,\varphi$; свободные вхождения остальных переменных в φ являются свободными вхождениями в эти две формулы.

Сравнивая это определение с индуктивным определением параметров формулы в разделе 5, мы видим, что параметры — это переменные, имеющие свободные вхождения в формулу.

Вхождения переменной, не являющиеся свободными (в том числе стоящие рядом с квантором) называют *связанными*. Например, переменная x имеет одно свободное и три связанных вхождения в формулу $A(x) \wedge \exists x\ B(x, x)$.

Теперь можно внести поправку в сказанное выше и считать, что аксиомами являются формулы

$$\forall\xi\,\varphi \to \varphi(t/\xi),$$

где $\varphi(t/\xi)$ есть результат подстановки t вместо всех *свободных* вхождений переменной ξ. Однако такой оговорки недостаточно, как показывает следующий пример.

Подставляя $f(y)$ вместо x в формулу $\forall z\, B(x,z)$, мы получаем (в полном согласии с нашей интуицией) формулу $\forall z\, B(f(y),z)$. Теперь рассмотрим формулу $\forall y\, B(x,y)$, которая отличается от $\forall z\, B(x,z)$ лишь именем связанной переменной и должна иметь тот же смысл. Переменная x в ней по-прежнему свободна, но подстановка $f(y)$ вместо x даёт формулу $\forall y\, B(f(y),y))$, в которой $f(y)$ неожиданно для себя попадает в область действия квантора по y. Такое явление иногда называют *коллизией переменных*; при этом подстановка даёт формулу, имеющую совсем не тот смысл, какой мы хотели.

Пример формулы вида $\forall\xi\,\varphi \to \varphi(t/\xi)$, в которой происходит коллизия переменных и которая не является общезначимой: $\forall x\,\exists y\, A(x,y) \to \exists y A(y,y)$.

Поэтому нам придётся принять ещё одну меру предосторожности и формально определить понятие *корректной* подстановки терма вместо переменной. Мы будем говорить, что подстановка терма t вместо переменной ξ корректна, если в процессе текстуальной замены всех свободных вхождений переменной ξ на терм t никакая переменная из t не попадает в область действия одноимённого квантора.

Главная часть в этом определении, во-первых, что вместо связанных вхождений переменных ничего подставлять не надо, а во-вторых, чтобы при корректной подстановке переменные из терма t не подпадали под действие одноимённых кванторов.

После всех этих приготовлений мы можем сформулировать две оставшиеся схемы аксиом исчисления предикатов: формула

(12) $\forall\xi\,\varphi \to \varphi(t/\xi)$

и двойственная ей формула

(13) $\varphi(t/\xi) \to \exists\xi\,\varphi$

будут аксиомами исчисления предикатов, если указанные в них подстановки корректны.

Два частных случая, когда подстановка заведомо корректна: во-первых, можно безопасно подставлять константу (или любой терм без параметров), во-вторых, подстановка переменной вместо себя всегда корректна (и ничего не меняет в формуле).

Отсюда следует, что формулы $\forall\xi\,\varphi \to \varphi$ и $\varphi \to \exists\xi\,\varphi$ будут аксиомами исчисления предикатов (для любой формулы φ и переменной ξ).

Нужны ли нам ещё какие-нибудь аксиомы и правила вывода? Конечно, нужны, поскольку уже сформулированные аксиомы не полностью отражают смысл кванторов. (Например, они вполне согласуются с таким пониманием этого смысла: формула $\forall\xi\,\varphi$ всегда ложна, а формула $\exists\xi\,\varphi$ всегда истинна.) Поэтому мы введём в наше исчисление два правила вывода, называемые *правилами Бернайса*, и на этом определение исчисления предикатов будет завершено.

Если переменная ξ не является параметром формулы ψ, то правила Бернайса разрешают такие переходы:

$$\frac{\psi\to\varphi}{\psi\to\forall\xi\,\varphi}\qquad\qquad\frac{\varphi\to\psi}{\exists\xi\,\varphi\to\psi}$$

Мы говорим, что стоящая снизу от черты (в каждом из правил) формула получается по соответствующему правилу из верхней. Соответственно дополняется и определение вывода как последовательности формул, в которой каждая формула либо является аксиомой, либо получается из предыдущих по одному из правил вывода (раньше было только правило MP, теперь добавились два новых правила).

Поясним интуитивный смысл этих правил. Первое говорит, что если из ψ следует φ, причём в φ есть параметр ξ, которого нет в ψ, то это означает, что формула φ истинна при всех значениях параметра ξ, если только формула ψ истинна.

Используя первое правило Бернайса, легко установить допустимость *правила обобщения*

$$\frac{\varphi}{\forall\xi\,\varphi}\qquad\text{(Gen)}$$

(если в исчислении предикатов выводима формула сверху от черты, то выводима и формула снизу). В самом деле, возьмём какую-нибудь выводимую формулу ψ без параметров (например, аксиому, в которой вместо A, B и C подставлены замкнутые формулы). Раз выводима формула φ, то выводима и формула $\psi\to\varphi$ (поскольку $\varphi\to(\psi\to\varphi)$ является тавтологией и даже аксиомой). Теперь по правилу Бернайса выводим $\psi\to\forall\xi\,\varphi$ и применяем правило MP к этой формуле и к формуле ψ.

Правило (Gen) (от Generalization — обобщение) кодифицирует стандартную практику рассуждений: мы доказываем какое-то утверждение φ со свободной переменной ξ, после чего заключаем, что мы доказали $\forall\xi\,\varphi$, так как ξ было произвольным.

Второе правило Бернайса также вполне естественно: желая доказать ψ в предположении $\exists\xi\,\varphi$, мы говорим: пусть такое ξ существует, возьмём его и докажем ψ (то есть докажем $\varphi \to \psi$ со свободной переменной ξ).

Как и в случае исчисления высказываний, перед нами стоят две задачи: надо доказать корректность исчисления предикатов (всякая выводимая формула общезначима) и его полноту (всякая общезначимая формула выводима). Этим мы и займёмся в следующих разделах.

§11. Корректность исчисления предикатов

Теорема 22. *Всякая выводимая в исчислении предикатов формула является общезначимой.*

Доказательство. Для исчисления высказываний проверка корректности была тривиальной — надо было по таблице проверить, что все аксиомы (1)-(11) являются тавтологиями. С этими аксиомами и сейчас нет проблем. Но в двух следующих аксиомах есть ограничение на корректность подстановки, без которого они могут не быть общезначимыми. Это ограничение должно быть использовано и в доказательстве корректности, что требует длинныхных рассуждений — при том, что сам факт кажется ясным и так; это доказательство мы пропустим.

□

§12. Полнота исчисления предикатов

В этом разделе мы докажем, что всякая общезначимая формула выводима в исчислении предикатов. Введём понятия непротиворечивой и полной теории.

Фиксируем некоторую сигнатуру σ. Пусть Γ — теория в сигнатуре σ, то есть произвольное множество замкнутых формул этой сигнатуры. Говорят, что теория Γ *противоречива*, если в ней выводится некоторая формула φ и её отрицание $\neg\varphi$. В этом случае из Γ выводится любая формула, так как имеется аксиома $\neg A \to (A \to B)$. Если теория Γ не является противоречивой, то она называется *непротиворечивой*.

Заметим, что теория противоречива тогда и только тогда, когда в ней выводится формула $\varphi \wedge \neg\varphi$ (здесь φ — произвольная формула сигнатуры).

Непосредственно из определения следует, что всякое подмножество непротиворечивого множества непротиворечиво. Кроме того, если бесконечное множество противоречиво, то некоторое его конечное подмножество тоже противоречиво (поскольку в выводе участвует лишь конечное число формул).

Синтаксическое понятие непротиворечивости мы будем сравнивать с семантическим понятием совместности. Пусть имеется некоторая интерпретация M сигнатуры σ. Напомним, что она называется *моделью* теории Γ, если все формулы из Γ истинны в M. Множество Γ называется *совместным*, если оно имеет модель, то есть если все его формулы истинны в некоторой интерпретации.

Теорема 23 (о корректности; переформулировка). *Любое совместное множество замкнутых формул непротиворечиво.*

Доказательство. Пусть все формулы из Γ истинны в некоторой интерпретации M. Может ли оказаться, что $\Gamma \vdash \varphi$ и $\Gamma \vdash \neg\varphi$ для некоторой замкнутой формулы φ? Легко понять, что нет. В самом деле, в этом случае формулы φ и $\neg\varphi$ должны быть одновременно истинны в M, что, очевидно, невозможно. □

Для доказательства обратного утверждения (о совместности непротиворечивой теории) нам понадобится понятие полной теории.

Непротиворечивое множество Γ, состоящее из замкнутых формул сигнатуры σ, называется *полным* в этой сигнатуре, если для любой замкнутой формулы φ этой сигнатуры либо формула φ, либо её отрицание $\neg\varphi$ выводятся из Γ.

Другими словами, теория полна, если из любых двух формул φ и $\neg\varphi$ (соответствующей сигнатуры) ровно одна является теоремой этой теории.

Полное множество можно получить, взяв какую-либо интерпретацию и рассмотрев все замкутые формулы, истинные в этой интерпретации. (Впоследствии мы увидим, что любое полное множество может быть получено таким способом — это легко следует из теоремы 24.)

В определении полноты существенно, что мы ограничиваемся замкнутыми формулами той же сигнатуры. Например, если мы возьмём одноместный предикатный символ S, не входящий в Γ, то формулы из Γ про него ничего не утверждают, и потому, скажем, ни формула $\forall x\, S(x)$, ни её отрицание не выводимы из Γ. Замкнутость формулы φ тоже важна. Например, множество всех истинных в натуральном ряду формул сигнатуры $(=, <)$ полно, но ни формула $x = y$, ни формула $x \neq y$ из него не выводятся, иначе по правилу обобщения мы получили бы ложную в $\mathbb{N}$ формулу $\forall x \forall y\, (x = y)$ или $\forall x \forall y\, (x \neq y)$.

Теперь мы готовы к формулировке основного результата этого раздела.

Теорема 24 (полнота исчисления предикатов, сильная форма). *Любая непротиворечивая теория совместна.*

Доказательство сводится к проверке формул и здесь не приводится. Возвратимся теперь к формулировке теоремы о полноте.

Теорема 25 (полнота исчисления предикатов, слабая форма). *Всякая общезначимая формула выводима в исчислении предикатов.*

Доказательство. Пусть формула φ замкнута. Если она невыводима, то множество $\{\neg\varphi\}$ непротиворечиво и потому совместно. В его модели формула φ будет ложной, что противоречит предположению.

Что касается незамкнутых формул, то их общезначимость и выводимость равносильна общезначимости и выводимости их замыкания. $\square$

Глава III

Теория алгоритмов

§1. Вычислимые функции

Функция f с натуральными аргументами и значениями называется *вычислимой*, если существует алгоритм, её вычисляющий, то есть такой алгоритм A, что

- если $f(n)$ определено для некоторого натурального n, то алгоритм A останавливается на входе n и печатает $f(n)$;
- если $f(n)$ не определено, то алгоритм A не останавливается на входе n.

Несколько замечаний по поводу этого определения:

1. Понятие вычислимости определяется здесь для частичных функций (областью определения которых является некоторое подмножество натурального ряда). Например, нигде не определённая функция вычислима (в качестве A надо взять программу, которая всегда зацикливается).

2. Можно было бы изменить определение, сказав так: "если $f(n)$ не определено, то либо алгоритм A не останавливается, либо останавливается, но ничего не печатает". На самом деле от этого ничего бы не изменилось (вместо того, чтобы останавливаться, ничего не напечатав, алгоритм может зацикливаться).

3. Входами и выходами алгоритмов могут быть не только натуральные числа, но и двоичные строки (слова в алфавите $\{0,1\}$), пары натуральных чисел, конечные последовательности слов и вообще любые, как говорят, "конструктивные объекты". Поэтому аналогичным образом можно определить понятие, скажем, вычислимой функции с двумя натуральными аргументами, значениями которой являются рациональные числа.

Для функций, скажем, с действительными аргументами и значениями понятие вычислимости требует специального определения. Здесь ситуация сложнее, определения могут быть разными, и мы о вычислимости таких функций говорить не будем. Отметим только, что, например, синус (при разумном определении вычислимости) вычислим, а функция $\operatorname{sign}(x)$, равная -1, 0 и 1 при $x < 0$, $x = 0$ и $x > 0$ соответственно — нет. Точно так же требует специального определения вычислимость функций, аргументами которых являются бесконечные последовательности нулей и единиц и т.п.

4. Несколько десятилетий назад понятие алгоритма требовало специального разъяснения. Сейчас ("компьютерная грамотность"?) такие объяснения всё равно никто читать не будет, поскольку и так ясно, что такое алгоритм. Но всё же надо соблюдать осторожность, чтобы не принять за алгоритм то, что им не является. Вот пример неверного рассуждения:

"Докажем что всякая вычислимая функция f с натуральными аргументами и значениями может быть продолжена до всюду определённой вычислимой функции $g\colon \mathbb{N} \to \mathbb{N}$. В самом деле, если f вычисляется алгоритмом A, то следующий алгоритм B вычисляет функцию g, продолжающую f: "если A останавливается на n, то B даёт тот же результат, что и A; если A не останавливается на n, то B даёт результат (скажем) 0". (В чём ошибка в этом рассуждении?)

§2. Разрешимые множества

Множество натуральных чисел X называется *разрешимым*, если существует алгоритм, который по любому натуральному n определяет, принадлежит ли оно множеству X.

Другими словами, X разрешимо, если его *характеристическая функция* $\chi(n) = (\textbf{if n} \in \textbf{X then 1 else 0 fi})$ вычислима.

Очевидно, пересечение, объединение и разность разрешимых множеств разрешимы. Любое конечное множество разрешимо.

Аналогично определяют разрешимость множеств пар натуральных чисел, множеств рациональных чисел и т.п.

Можно доказать, что непустое множество натуральных чисел разрешимо тогда и только тогда, когда оно есть множество значений всюду определённой неубывающей вычислимой функции с натуральными аргументами и значениями.

Существуют ли неразрешимые множества? Существуют — просто потому, что алгоритмов (и поэтому разрешимых подмножеств натурального ряда) счётное число, а всех подмножеств натурального ряда несчётное число. Более конкретные примеры мы ещё построим.

§3. Перечислимые множества

Множество натуральных чисел называется *перечислимым*, если оно перечисляется некоторым алгоритмом, то есть если существует алгоритм, который печатает (в произвольном порядке и с произвольными промежутками времени) все элементы этого множества и только их.

Такой алгоритм не имеет входа; напечатав несколько чисел, он может надолго задуматься и следующее число напечатать после большого перерыва (а может вообще больше никогда ничего не напечатать — тогда множество будет конечным).

Существует много эквивалентных определений перечислимого множества. Вот некоторые из них:

(1) Множество перечислимо, если оно есть область определения вычислимой функции.
(2) Множество перечислимо, если оно есть область значений вычислимой функции.
(3) Множество X перечислимо, если его (как иногда говорят) "полухарактеристическая" функция, равная 0 на элементах X и не определённая вне X, вычислима.

Чтобы доказать эквивалентность этих определений, воспользуемся возможностью пошагового исполнения алгоритма.

Пусть X перечисляется некоторым алгоритмом A. Покажем, что полухарактеристическая функция множества X вычислима. В самом деле, алгоритм её вычисления таков: получив на вход число n, пошагово выполнять алгоритм A, ожидая, пока он напечатает число n. Как только он это сделает, выдать на выход 0 и закончить работу.

Наоборот, пусть X есть область определения (вычислимой) функции f, вычисляемой некоторым алгоритмом B. Тогда X перечисляется таким алгоритмом A: ßПараллельно запускать B на входах $0, 1, 2, \ldots$, делая всё больше шагов работы алгоритма B (сначала один шаг работы на входах 0 и 1; потом по два шага работы на входах $0, 1, 2$, потом по три на входах $0, 1, 2, 3$ и так далее). Все аргументы, на которых алгоритм B заканчивает работу, печатать по мере обнаружения.

Итак, мы установили эквивалентность исходного определения определениям 1 и 3. Если в только что приведённом описании алгоритма A печатать не аргументы, на которых B заканчивает работу, а результаты этой работы, то получается алгоритм, перечисляющий область значений функции f. Осталось ещё убедиться, что всякое перечислимое множество есть область значений вычислимой функции. Это можно сделать, например, так: пусть X есть область определения вычислимой функции, вычисляемой некоторым алгоритмом A. Тогда X есть область значений функции

$$b(x) = \begin{cases} x, \text{ если } A \text{ заканчивает работу на } x, \\ \text{не определено в противном случае.} \end{cases}$$

Вычисляющий эту функцию алгоритм действует так же, как и A, но только вместо результата работы алгоритма A выдаёт копию входа.

Ещё одно эквивалентное определение перечислимого множества: множество натуральных чисел перечислимо, если оно либо пусто, либо есть множество значений всюду определённой вычислимой функции (другими словами, его элементы можно расположить в вычислимую последовательность).

В самом деле, пусть перечислимое множество X, перечисляемое алгоритмом A, непусто. Возьмём в нём какой-то элемент x_0. Теперь рассмотрим такую всюду определённую функцию a: если на n-м шаге работы алгоритма A появляется число t, то положим $a(n) = t$; если же ничего не появляется, то положим $a(n) = x_0$. (Мы предполагаем, что на данном шаге работы алгоритма может появиться только одно число — в противном случае работу надо разбить на более мелкие шаги.)

Заметим, что это рассуждение неконструктивно — имея алгоритм A, мы можем не знать, пусто ли перечисляемое им множество или нет.

Теорема 26. *Пересечение и объединение перечислимых множеств перечислимы.*

Доказательство. Если X и Y перечисляются алгоритмами A и B, то их объединение перечисляется алгоритмом, который параллельно выполняет по шагам A и B и печатает всё, что печатают A и B. С пересечением немного сложнее — результаты работы A и B надо накапливать и сверять друг с другом; что появится общего — печатать. □

Как мы увидим, дополнение перечислимого множества не обязано быть перечислимым.

§4. Перечислимые и разрешимые множества

Теорема 27. *Всякое разрешимое множество натуральных чисел перечислимо. Если множество A и его дополнение (до множества всех натуральных чисел) перечислимы, то A разрешимо.*

Доказательство. Если принадлежность числа к множеству A можно проверить некоторым алгоритмом, то A и его дополнение перечислимы: надо по очереди проверять принадлежность чисел $0, 1, 2, \ldots$ и печатать те из них, которые принадлежат A (или те, которые не принадлежат A).

В другую сторону: если у нас есть алгоритм, перечисляющий A, а также другой алгоритм, перечисляющий дополнение к A, то для выяснения принадлежности заданного числа n к A надо запустить оба эти алгоритма и ждать, пока один из них напечатает n (мы знаем, что рано или поздно

ровно один из них это сделает). Посмотрев, какой алгоритм это сделал, мы узнаем, лежит ли n в A. □

Этот факт называют *теоремой Пóста*.

Она говорит, что разрешимые множества — это перечислимые множества с перечислимыми дополнениями. Напротив, перечислимые множества можно определить через разрешимые:

Теорема 28. *Множество P натуральных чисел перечислимо тогда и только тогда, когда оно является проекцией некоторого разрешимого множества Q пар натуральных чисел. (Проекция получается, если от пар оставить их первые компоненты: $x \in P \Leftrightarrow \exists y(\langle x, y\rangle \in Q)$.)*

Доказательство. Проекция любого перечислимого множества перечислима (перечисляющий алгоритм должен лишь удалять вторые члены пар), так что проекция разрешимого множества тем более перечислима.

Напротив, если P — перечислимое множество, перечисляемое алгоритмом A, то оно есть проекция разрешимого множества Q, состоящего из всех таких пар $\langle x, n\rangle$, что x появляется в течении первых n шагов работы алгоритма A. (Это свойство, очевидно, разрешимо.) □

§5. Перечислимость и вычислимость

Мы видели, что перечислимое множество можно определить в терминах вычислимых функций (например, как область определения вычислимой функции). Можно сделать и наоборот:

Теорема 29. *Функция f с натуральными аргументами и значениями вычислима тогда и только тогда, когда её график*

$$F = \{\langle x, y\rangle \mid f(x) \text{ определено и равно } y\}$$

является перечислимым множеством пар натуральных чисел.

Доказательство. Пусть f вычислима. Тогда существует алгоритм, перечисляющий её область определения, то есть печатающий все x, на которых f определена. Если теперь для каждого из таких x вычислять ещё и значение $f(x)$, получим алгоритм, перечисляющий множество F.

Напротив, если имеется алгоритм, перечисляющий F, то функция f вычисляется таким алгоритмом: имея на входе n, ждём появления в F пары, первый член которой равен n; как только такая пара появилась, печатаем её второй член и кончаем работу. □

Пусть f — частичная функция с натуральными аргументами и значениями. *Образ* множества A при f определяется как множество всех чисел $f(n)$, для которых $n \in A$ и $f(n)$ определено. *Прообраз* множества A при f определяется как множество всех тех n, при которых $f(n)$ определено и принадлежит A.

Теорема 30. *Прообраз и образ перечислимого множества при вычислимой функции перечислимы.*

Доказательство. В самом деле, прообраз перечислимого множества A при вычислимой функции f можно получить так: взять график f, пересечь его с перечислимым множеством $\mathbb{N} \times A$ и спроектировать на первую координату. Рассуждение для образов аналогично, только координаты меняются местами. □

§6. Зачем нужны простые вычислительные модели?

До сих пор нам было удобно ссылаться на программистский опыт, говоря об алгоритмах, программах, интерпретаторах, пошаговом выполнении и т.д. Это позволяло нам игнорировать детали построения тех или иных алгоритмов под тем предлогом, что читатель их легко восстановит (или хотя бы поверит — всё-таки не каждый читатель в своей жизни писал интерпретатор языка C на C).

Но в некоторых случаях этого недостаточно. Пусть, например, мы хотим доказать алгоритмическую неразрешимость какой-то задачи, в определении которой ничего не говорится о программах. Это обычно делается так. Мы показываем, что проблема остановки сводится к этой задаче. Для этого мы моделируем работу произвольного алгоритма в терминах рассматриваемой задачи (что это значит, будет видно из приводимого ниже примера). При этом нам важно, чтобы определение алгоритма было как можно проще.

Таким образом, наш план таков. Мы опишем довольно просто определяемый класс машин (его можно выбирать по-разному, мы будем использовать так называемые машины Тьюринга), затем объявим, что всякая вычислимая функция может быть вычислена на такой машине.

Другая причина, по которой важны простые вычислительные модели (таких моделей много — разные виды машин Тьюринга, адресные машины и т.п.), связана с теорией сложности вычислений, когда нас начинает интересовать время выполнения программ. Но этот вопрос выходит за рамки классической теории алгоритмов.

§7. Машины Тьюринга: определение

Машина Тьюринга имеет бесконечную в обе стороны *ленту*, разделённую на квадратики (*ячейки*). В каждой ячейке может быть записан некоторый символ из фиксированного (для данной машины) конечного множества, называемого *алфавитом* данной машины. Один из символов алфавита выделен и называется "пробелом— предполагается, что изначально вся лента пуста, то есть заполнена пробелами.

Машина Тьюринга может менять содержимое ленты с помощью специальной читающей и пишущей *головки*, которая движется вдоль ленты. В каждый момент головка находится в одной из ячеек. Машина Тьюринга получает от головки информацию о том, какой символ та видит, и в зависимости от этого (и от своего внутреннего состояния) решает, что делать, то есть какой символ записать в текущей ячейке и куда сдвинуться после этого (налево, направо или остаться на месте). При этом также меняется внутреннее состояние машины (мы предполагаем, что машина — не считая ленты — имеет конечную память, то есть конечное число внутренних состояний). Ещё надо договориться, с чего мы начинаем и когда кончаем работу.

Таким образом, чтобы задать машину Тьюринга, надо указать следующие объекты:

- произвольное конечное множество A (*алфавит*); его элементы называются *символами*;
- некоторый выделенный символ $a_0 \in A$ (*пробел*, или *пустой символ*);
- конечное множество S, называемое множеством *состояний*;
- некоторое выделенное состояние $s_0 \in S$, называемое *начальным*;
- *таблицу переходов*, которая определяет поведение машины в зависимости от состояния и текущего символа (см. ниже);
- некоторое подмножество $F \subset S$, элементы которого называются *заключительными состояниями* (попав в такое состояние, машина останавливается).

Таблица переходов устроена следующим образом: для каждой пары ⟨текущее состояние, текущий символ⟩ указана тройка ⟨новое состояние, новый символ, сдвиг⟩. Здесь сдвиг — одно из чисел -1 (влево), 0 (на месте) и 1 (направо). Таким образом, таблица переходов есть функция типа $S \times A \to S \times A \times \{-1, 0, 1\}$, определённая на тех парах, в которых состояние не является заключительным.

Остаётся описать поведение машины Тьюринга. В каждый момент имеется некоторая *конфигурация*, складывающаяся из содержимого ленты (формально говоря, содержимое ленты есть произвольное отображение $\mathbb{Z} \to A$),

текущей позиции головки (некоторое целое число) и текущего состояния машины (элемент S). Преобразование конфигурации в следующую происходит по естественным правилам: мы смотрим в таблице, что надо делать для данного состояния и для данного символа, то есть выясняем новое состояние машины, меняем символ на указанный и после этого сдвигаем головку влево, вправо или оставляем на месте. При этом, если новое состояние является одним из заключительных, работа машины заканчивается. Остаётся договориться, как мы подаём информацию на вход машины и что считается результатом её работы. Будем считать, что алфавит машины, помимо пробела, содержит символы 0 и 1 (а также, возможно, ещё какие-то символы). Входом и выходом машины будут конечные последовательности нулей и единиц (двоичные слова). Входное слово записывается на пустой ленте, головка машины ставится в его первую клетку, машина приводится в начальное состояние и запускается. Если машина останавливается, результатом считается двоичное слово, которое можно прочесть, начиная с позиции головки и двигаясь направо (пока не появится символ, отличный от 0 и 1).

Таким образом, любая машина Тьюринга задаёт некоторую частичную функцию на двоичных словах. Все такие функции естественно назвать *вычислимыми на машинах Тьюринга*.

§8. Машины Тьюринга: обсуждение

Разумеется, наше определение содержит много конкретных деталей, которые можно было бы изменить. Например, лента может быть бесконечной только в одну сторону. Можно придать машине две ленты. Можно считать, что машина может либо написать новый символ, либо сдвинуться, но не то и другое вместе. Можно ограничить алфавит, считая, скажем, что в нём должно быть ровно 10 символов. Можно потребовать, чтобы в конце на ленте ничего не было, кроме результата работы (остальные клетки должны быть пусты). Все перечисленные и многие другие изменения не меняют класса вычислимых на машинах Тьюринга функций. Конечно, есть и небезобидные изменения. Например, если запретить машине двигаться налево, то это радикально поменяет дело — по существу лента станет бесполезной, так как к старым записям уже нельзя будет вернуться.

Как понять, какие изменения безобидны, а какие нет? Видимо, тут необходим некоторый опыт практического программирования на машинах Тьюринга, хотя бы небольшой. После этого уже можно представлять себе возможности машины, не выписывая программы полностью, а руководствуясь лишь приблизительным описанием. В качестве примера опишем машину,

которая удваивает входное слово (изготавливает слово XX, если на входе было слово X).

Если машина видит пробел (входное слово пусто), она кончает работу. Если нет, она запоминает текущий символ и ставит пометку (в алфавите помимо символов 0 и 1 будут ещё их "помеченные варианты-$\overline{0}$ и $\overline{1}$). Затем она движется направо до пустой клетки, после чего пишет там копию запомненного символа. Затем она движется налево до пометки; уткнувшись в пометку, отходит назад и запоминает следующий символ и так далее, пока не скопирует всё слово.

Имея некоторый опыт, можно за всеми этими фразами видеть конкретные куски программы для машины Тьюринга. Например, слова "запоминает символ и движется направо" означают, что есть две группы состояний, одна для ситуации, когда запомнен нуль, другая — когда запомнена единица, и внутри каждой группы запрограммировано движение направо до первой пустой клетки.

Имея ещё чуть больше опыта, можно понять, что в этом описании есть ошибка — не предусмотрен механизм остановки, когда всё слово будет скопировано, поскольку копии символов ничем не отличаются от символов исходного слова. Ясно и то, как ошибку исправить — надо в качестве копий писать специальные символы $\tilde{0}$ и $\tilde{1}$, а на последнем этапе все пометки удалить.

Другой пример неформального рассуждения: объясним, почему можно не использовать дополнительных символов, кроме 0, 1 и пустого символа. Пусть есть машина с большим алфавитом из N символов. Построим новую машину, которая будет моделировать работу старой, но каждой клетке старой будет соответствовать блок из k клеток новой. Размер блока (число k) будет фиксирован так, чтобы внутри блока можно было бы закодировать нулями и единицами все символы большого алфавита. Исходные символы 0, 1 и пустой будем кодировать как 0, за которым идут $(k-1)$ пустых символов, 1, за которым идут $(k-1)$ пустых символов, и группу из k пустых символов. Для начала надо раздвинуть буквы входного слова на расстояние k, что можно сделать без дополнительных символов (дойдя до крайней буквы, отодвигаем её, затем дойдя до следующей, отодвигаем её и крайнюю и так далее); надо только понимать, что можно идентифицировать конец слова как позицию, за которой следует более k пустых символов. Ясно, что в этом процессе мы должны хранить в памяти некоторый конечный объём информации, так что это возможно. После этого уже можно моделировать работу исходной машины по шагам, и для этого тоже достаточно конечной памяти (конечного числа состояний), так как нам важна только небольшая

окрестность головки моделируемой машины. Наконец, надо сжать результат обратно.

Утверждение о том, что всякая вычислимая функция вычислима на машине Тьюринга, называют *тезисом Тьюринга*. Конечно, его смысл зависит от того, что понимать под словами "вычислимая функция". Если понимать их в расплывчато-интуитивном смысле ("функция вычисляется алгоритмически, то есть по чётким, недвусмысленным, однозначным правилам" или что-то в таком роде), конечно, ни о каком доказательстве тезиса Тьюринга не может быть речи. Можно лишь говорить, что многовековая практика человечества от Евклида до Кнута (ныне живущий крупнейший специалист в теории алгоритмических языков) не встретилась с примером алгоритма, который нельзя было бы записать как программу машины Тьюринга и т.п. Впрочем, ещё один (не слишком убедительный) аргумент приведён ниже.

Но если понимать слово "вычислимая" в тезисе Тьюринга как "вычислимая с помощью программы на C" и представить себе на минуту, что синтаксис и семантика C-программ точно определены, то тезис Тьюринга станет уже чётким утверждением, которое может быть истинным или ложным, и которое можно доказывать. Конечно, такое доказательство по необходимости должно использовать формальное описание синтаксиса и семантики C, и потому никем не проводилось, но для более простых вычислительных моделей это действительно можно формально доказать. Впрочем, такого рода доказательства сродни доказательству корректности длинной программы, и потому желающих их писать и тем более читать немного.

В заключении приведём обещанный аргумент в пользу того, что любая вычислимая функция вычислима на машине Тьюринга. Пусть есть функция, которую человек умеет вычислять. При этом, он, естественно, должен использовать карандаш и бумагу, так как количество информации, которое он может хранить "в уме ограничено. Будем считать, что он пишет на отдельных листах бумаги. Помимо текущего листа, есть стопка бумаг справа и стопка слева; в любую из них можно положить текущий лист, завершив с ним работу, а из другой стопки взять следующий. У человека есть карандаш и ластик. Поскольку очень мелкие буквы на листе неразличимы, число отчётливо различных состояний листа конечно, так что можно считать, что в каждый момент на листе записана одна буква из некоторого конечного (хотя и весьма большого) алфавита. Человек тоже имеет конечную память, так что его состояние есть элемент некоторого конечного множества. При этом можно составить некоторую таблицу, в которой записано, чем кончится его работа над листом с данным содержимым, начатая в данном состоянии (что будет на листе, в каком состоянии будет человек и из какой пачки

будет взят следующий лист). Теперь уже видно, что действия человека как раз соответствуют работе машины Тьюринга с большим (но конечным) алфавитом и большим (но конечным) числом внутренних состояний.

§9. Программы с конечным числом переменных

Мы хотим показать, что график всякой вычислимой функции является арифметическим множеством, то есть выразим формулой арифметики. Для этого удобно перейти от машин Тьюринга к другой модели, которую можно условно назвать машинами с конечным числом регистров.

Программа для такой машины использует конечное число переменных, значениями которых являются натуральные числа. Числа эти могут быть произвольного размера, так что машина реально имеет память неограниченного объёма. Программа состоит из нумерованных по порядку команд. Каждая команда имеет один из следующих видов:

- `a=0;`
- `a=b;`
- `a=b+1;`
- `a=b-1;`
- `goto met;`
- `if (a==0) goto met1; else goto met2;`
- `exit(0);`

Поскольку мы считаем, что значения переменных — натуральные (целые неотрицательные) числа, условимся считать разность $0 - 1$ равной 0 (впрочем, это не так важно — можно было бы считать это аварией).

Дойдя до команды `exit`, программа заканчивает работу.

Как и для машин Тьюринга, полезна некоторая практика программирования. Для тренировки напишем программу сложения двух чисел. Она помещает в `c` сумму чисел, которые были в переменных `a` и `b`. Такая программа на C имела бы вид

```
c=a;
/* инвариант: ответ = сумма текущих значений c и b */
while (b!=0)
{
   c++;
   b--;
}
```

Имитируя цикл с помощью операторов перехода, получаем программу для нашей машины:

```
1: c=a;
2: if (b==0) goto 6; else goto 3;
3: c++;
4: b--;
5: goto 2;
6: exit(0);
```

Теперь легко понять, как написать программы для вычитания, умножения (которое реализуется как цикл с повторным сложением), деления с остатком (деление есть сокращённое вычитание), возведения в степень, проверки простоты, отыскания n-го простого числа и т.п. Вообще по сравнению с машинами Тьюринга этот язык более привычен и потому легче поверить, что на нём можно запрограммировать все алгоритмы.

Единственное, чего в нём реально не хватает — это массивов. Но это легко обойти, поскольку есть числа произвольного размера и нас не интересует число операций (как это принято в общей теории алгоритмов). Вместо массива битов мы можем хранить число, двоичной записью которого он является, а для массивов чисел воспользоваться, скажем, основной теоремой арифметики и хранить последовательность $\langle a, b, c, d, e\rangle$ как число $2^a3^b5^c7^d11^e$. При этом операции `a[i]=b` и `b=a[i]` заменяются на небольшие программы, которые содержат переменные `a`, `b`, `i` и ещё несколько переменных. (Частью этих программ является нахождение простого числа с заданным порядковым номером.)

Легко определить понятие вычислимой (в этой модели) функции. Пусть есть программа с двумя переменными `x` и `y` (и, `".',.&-.`, другими). Поместим в `x` некоторое число n, а в остальные переменные поместим нули. Запустим программу. Если она не остановится, то вычисляемая ей функция в точке n не определена. Если остановится, то содержимое переменной `y` после остановки и будет значением функции, вычисляемой нашей программой (в точке n). Функция называется вычислимой (в этой модели), если существует вычисляющая её программа.

Как всегда, при определении мы фиксировали различные детали, большинство из которых не являются существенными. Можно было бы добавить некоторые команды (сложение, например) — или даже исключить (например, без копирования можно обойтись с помощью небольшой хитрости).

Несколько более удивительно, что число переменных можно ограничить, скажем, сотней — но и это не так уж странно, если вспомнить, что мы можем хранить в одной переменной целый массив.

§10. Машины Тьюринга и программы

Построенная вычислительная модель не слабее машин Тьюринга в том смысле, что любую вычислимую на машинах Тьюринга функцию можно вычислить и программой с конечным числом переменных.

Теорема 31. *Всякая функция, вычислимая на машинах Тьюринга, может быть вычислена с помощью программы описанного вида с конечным числом переменных.*

Следует уточнить, однако, что мы имеем в виду, так как для машин Тьюринга исходное данное и результат были двоичными словами, а для программ — натуральными числами. Мы отождествляем те и другие по естественному правилу, при котором слова Λ (пустое), 0, 1, 00, 01,... соответствуют числам 0, 1, 2, 3, 4... (чтобы получить из числа слово, прибавим к нему единицу, переведём в двоичную систему и отбросим единицу в старшем разряде).

Доказательство. Как и раньше, мы приведём лишь приблизительное описание того, как по машине Тьюринга строится программа с конечным числом переменных, вычисляющая ту же функцию. Прежде всего конфигурации машины Тьюринга надо закодировать числами. Можно сделать это, например, поставив в соответствие каждой конфигурации четыре числа: номер текущего состояния, номер текущего символа (в ячейке, где стоит головка машины), код содержимого ленты слева от головки и код содержимого ленты справа от головки.

Чтобы решить, как удобнее кодировать содержимое ленты слева и справа от головки, заметим, что машина Тьюринга обращается с двумя половинами лентами слева и справа от головки как со стеками. (Стеком называется структура данных, напоминающая стопку листов. В неё можно положить лист наверх, взять верхний лист, а также проверить, есть ли ещё листы.) В самом деле, при движении головки направо из правого стека берётся верхний элемент, а в левый стек кладётся; при движении налево — наоборот. Стеки легко моделировать с помощью чисел: например, если в стеке хранятся символы 0 и 1, то добавление нуля соответствует операции $x \mapsto 2x$, добавление единицы — операции $x \mapsto 2x + 1$, верхний элемент есть остаток при делении на 2, а удаление верхнего элемента есть деление на 2 (с отбрасыванием остатка). Другими словами, мы воспринимаем двоичную запись числа как стек, вершина которого находится справа, у младшего разряда. Точно так же можно использовать n-ичную систему счисления и представить стек с n возможными символами в каждой позиции.

Теперь основной цикл машины Тьюринга можно записать как программу, оперирующую с указанными четырьмя числами (символ, состояние, левый стек и правый стек) — без особых хитростей. Но несколько вещей всё-таки надо иметь в виду.

Во-первых, стеки конечны, а лента бесконечна — мы должны договориться, что если стек опустошается, то в него автоматически добавляется символ пробела. Тем самым бесконечный пустой хвост ленты может присутствовать в стеке, как теперь говорят, виртуально.

Во-вторых, напомним, что мы договорились отождествлять двоичные слова (которые подаются на вход машины Тьюринга) и их коды (которые хранятся в переменных нашей программы). Поэтому, получив код входного слова, надо его разобрать по символам и положить эти символы один за другим в стек (основания систем счисления разные, так что просто так переписать это нельзя). Аналогичные проблемы возникают и при превращении выхода (части содержимого правого стека) в соответствующее число, но все они легко преодолимы, и мы не будем вдаваться в подробности. □

Верно и обратное утверждение:

Теорема 32. *Всякая функция, вычислимая программой с конечным числом переменных, вычислима на машине Тьюринга.*

Доказательство. Нам надо моделировать поведение программы с помощью машины Тьюринга. Будем считать, что значения переменных записаны на ленте (в двоичной системе) и разделены специальным разделительным символом. Тогда машина может найти любую переменную, идя от начала ленты и считая разделительные символы, сделать что-то с этой переменной и затем вернуться обратно в начало. (Нет необходимости записывать на ленте номер исполняемой команды, поскольку команд конечное число и машина может помнить номер текущей команды как часть своего состояния.) Операции прибавления и вычитания единицы также легко выполнимы в двоичной записи (если идти справа налево). Надо только иметь в виду, что размер числа может увеличиться, и тогда нужно для него освободить место, сдвинув все символы справа от головки на одну позицию. (При уменьшении нужно сдвинуть влево.) Ясно, что это также легко выполнить с помощью машины Тьюринга.

Если мы записываем числа в двоичной системе, то проблемы с перекодированием при вводе-выводе минимальны (надо лишь дописать нули для значений остальных переменных в начале работы и встать в нужное место ленты в конце). □

§11. Арифметичность вычислимых функций

Сейчас мы докажем, что функции, вычислимые программами с конечным числом переменных, арифметичны, то есть их графики являются арифметическими множествами. В этом разделе мы вновь предполагаем некоторое знакомство читателя с логическими обозначениями, и будем рассматривать *арифметические формулы*, содержащие переменные по натуральным числам, равенство, константы 0 и 1, операции сложения и умножения, логические связки (И, ИЛИ, НЕ) и кванторы "для всех" и "существует". Формально говоря, мы рассматриваем сигнатуру, содержащую единственный двуместный предикатный символ (равенство), две константы 0 и 1 и два двуместных функциональных символа (сложение и умножение). Говоря об истинности таких формул, мы имеем в виду их истинность в стандартной интерпретации, носителем которой является множество $\mathbb{N}$ натуральных чисел.

Множество $A \subset \mathbb{N}^k$ называется *арифметическим*, если существует арифметическая формула α с параметрами $x_1, \ldots, x_k$, которая его представляет в следующем смысле: $\langle n_1, \ldots, n_k \rangle \in A$ тогда и только тогда, когда формула α истинна при значениях параметров $x_1 = n_1, \ldots, x_k = n_k$.

Теорема 33. *График любой функции, вычисляемой программой с конечным числом переменных, является арифметическим множеством.*

Доказательство. Пусть $f\colon \mathbb{N} \to \mathbb{N}$ — функция, вычислимая некоторой программой P с конечным числом переменных $k_1, \ldots, k_N$. Будем считать, что входной переменной является k_1, а выходной — k_2. Нам нужно написать формулу с двумя переменными x, y, которая была бы истинна тогда и только тогда, когда $y = f(x)$. Состояние программы с конечным числом переменных полностью описывается значениями переменных и номером текущей команды (в процессорах соответствующий регистр часто называют program counter, по-русски счётчик команд). Легко понять, что соответствие между двумя последовательными состояниями программы с конечным числом переменных арифметично. Мы имеем в виду, что можно написать арифметическую формулу

$$\mathrm{Step}(s_1, \ldots, s_N, p, s'_1, \ldots, s'_N, p'),$$

с $2N + 2$ переменными, которая утверждает, что данная программа P из состояния, где переменные равны $s_1, \ldots, s_N$, а счётчик команд равен p, за один шаг переходит в состояние, где переменные равны $s'_1, \ldots, s'_n$, а счётчик команд равен p'. (Договоримся, что значение $p' = 0$ соответствует остановке

программы.) Такая формула является конъюнкцией отдельных утверждений, соответствующих каждой строке программы. Пусть, например, строка 7 программы имеет вид $\mathtt{k}_2\mathtt{=k}_3$. Тогда в конъюнкции будет член вида

$$(p = 7) \Rightarrow ((s'_1 = s_1) \wedge (s'_2 = s_3) \wedge (s'_3 = s_3) \wedge \ldots \wedge (s'_N = s_N) \wedge (p' = 8)).$$

Для строки с условными переходами типа

```
3:  if (k5==0) goto 17; else goto 33;
```

в формуле будет два конъюнктивных члена (на два случая перехода)

$$((p = 3) \wedge (s_5 = 0)) \Rightarrow ((s'_1 = s_1) \wedge \ldots \wedge (s'_N = s_N) \wedge (p' = 17))$$

и

$$((p = 3) \wedge (s_5 \neq 0)) \Rightarrow ((s'_1 = s_1) \wedge \ldots \wedge (s'_N = s_N) \wedge (p' = 33)).$$

Надо ещё добавить утверждение о том, что при $p = 0$ работа прекращается, то есть что переменные на следующем шаге сохраняют свои значения, и p' остаётся равным 0.

Таким образом, арифметичность одного шага работы программы доказать несложно. Остаётся главный вопрос: как записать в виде формулы тот факт, что существует *последовательность* шагов, которая начинается с исходного состояния, заканчивается в данном и в которой каждый шаг правилен. Трудность в том, что здесь нужно как бы написать переменное число кванторов существования — или квантор "существует конечная последовательность натуральных чисел".

Это делается с помощью приёма, традиционно называемого β-функцией Гёделя. Вот что имеется в виду.

Лемма 1. Для любого k можно найти сколь угодно большое целое положительное число b, при котором первые k членов последовательности $b + 1, 2b + 1, 3b + 1, \ldots$ попарно взаимно просты.

Доказательство. Любой общий простой делитель двух из этих чисел будет делителем их разности, то есть числа lb при $0 < l < k$; взяв b кратным $k!$, мы гарантируем, что он будет делителем числа b, но все члены нашей последовательности взаимно просты с b. Лемма доказана.

Лемма 2. Для любой последовательности $x_0, x_1, \ldots, x_n$ натуральных чисел можно найти такие числа a и b, что x_i есть остаток от деления a на $b(i + 1) + 1$.

Доказательство. Согласно предыдущей лемме, делители $b(i+1)+1$ можно взять взаимно простыми (и сколь угодно большими), остаётся воспользоваться "китайской теоремой об остатках". Эта теорема утверждает, что если целые положительные числа $d_1, \ldots, d_k$ взаимно просты, то при делении целого u на них может получиться любой заданный набор остатков. В самом деле, таких наборов будет $d_1 d_2 \ldots d_k$ (поскольку при делении на d_i возможны

остатки от 0 до $d_i - 1$). При делении чисел $u = 0, 1, \ldots, d_1 d_2 \ldots d_k - 1$ получаются разные наборы остатков (если два числа u' и u'' дают одинаковые остатки, то их разность делится на все d_i, что невозможно в силу взаимной простоты). Поэтому чисел столько же, сколько наборов остатков, и должны появиться все наборы. Лемма 2 доказана.

Эта лемма показывает, что последовательность произвольной длины можно закодировать тремя числами a, b и n (последнее число — длина последовательности). Таким образом, условно говоря, можно заменить "формулу"

$$\exists \langle x_0, \ldots, x_n \rangle (\forall i \le n)[\ldots x_i \ldots]$$

(которая на самом деле не является арифметической формулой, так как содержит квантор по конечным последовательностям) на формулу

$$\exists a\, \exists b\, \exists n\, (\forall i \le n)[\ldots (\text{остаток от деления } a \text{ на } b(i+1)+1) \ldots].$$

Мы будем записывать остаток от деления a на $b(i+1)+1$ как $\beta(a, b, i)$ (отсюда и название "бета-функция").

Возвращаясь к программе P с конечным числом переменных $k_1, \ldots, k_N$ и вычисляемой ей функции f, можно записать утверждение вида $f(x) = y$ так: существуют такое число шагов n и такие числа $a_1, b_1, a_2, b_2, \ldots, a_N, b_N, a, b$, что

- $\beta(a_1, b_1, 0), \ldots, \beta(a_N, b_N, 0)$ есть правильные начальные значения переменных (первое равно x, остальные равны 0); $\beta(a, b, 0)$ есть правильное начальное значения счётчика команд, то есть 1;
- для каждого i от 0 до $n - 1$ имеет место

$$\mathrm{Step}(\beta(a_1, b_1, i), \ldots, \beta(a_N, b_N, i), \beta(a, b, i), \\ \beta(a_1, b_1, i+1), \ldots, \beta(a_N, b_N, i+1), \beta(a, b, i+1)),$$

  то есть каждый переход соответствует программе;
- $\beta(a_2, b_2, n) = y$ (значение выходной переменной k_2 в конце вычисления равно y) и $\beta(a, b, n) = 0$ (значение счётчика команд в конце вычисления равно 0, что по нашей договорённости соответствует остановке машины).

Итак, арифметичность вычислимых (на машинах с конечным числом переменных) функций доказана. □

Вспоминая теорему 31, мы заключаем, что всякая вычислимая на машине Тьюринга функция арифметична. Принимая тезис Тьюринга, можно сказать, что график любой вычислимой функции является арифметическим множеством.

§12. Теоремы Тарского и Гёделя

Поскольку графики вычислимых функций арифметичны, очевидно, разрешимые и перечислимые множества тоже будут арифметическими.

Рассмотрим теперь множество T, элементами которого являются все истинные арифметические формулы без параметров (точнее, их номера в какой-то вычислимой нумерации всех формул.)

Теорема 34. *Множество T не арифметично.*

Доказательство. Предположим, что множество номеров всех истинных арифметических формул (без параметров) арифметично. Пусть T — это множество, а $\tau(x)$ — соответствующая формула. Перенумеруем также все формулы с одним параметром x; пусть $F_n(x)$ — формула, имеющая номер n в этой нумерации. Рассмотрим формулу с единственным параметром x, утверждающую, что результат подстановки константы x в x-ую формулу с параметром ложен. Эту формулу можно написать так:

$$\exists z(\neg\tau(z) \wedge \mathrm{Subst}(z, x, x)),$$

где $\mathrm{Subst}(p, q, r)$ — формула с тремя параметрами, выражающая такое свойство: "p есть номер (в нумерации всех формул без параметров) той формулы, которая получится, если в q-ю формулу с одним параметром подставить константу r вместо этого параметра". Записанное в кавычках свойство описывает график некоторой вычислимой функции (соответствующей простым синтаксическим действиям и переходу от одной нумерации к другой) и потому существует выражающая его формула.

Итак, мы написали некоторую формулу с единственным параметром x. Пусть она имеет некоторый номер N. Подставим этот номер N вместо её параметра. Получится некоторая формула без параметров. Из построения видно, что эта формула истинна тогда и только тогда, когда результат подстановки числа N в формулу номер N (то есть сама эта формула!) ложен.

Это противоречие завершает доказательство теоремы Тарского. Мы видим, что нам потребовалась выразимость в арифметике не любых вычислимых функций, а одной вполне конкретной. При достаточном терпении соответствующую формулу можно-таки написать, и тем самым доказательство станет совсем "осязаемым".

□

Эта теорема называется *теоремой Тарского*. Её можно прочесть так: множество арифметических истин не арифметично. Или: понятие арифметической истины невыразимо в арифметике.

Теорема 35. *Множество T арифметических истин неперечислимо.*

Доказательство. В самом деле, любое перечислимое множество арифметично. □

Это утверждение называется *теоремой Гёделя о неполноте*. Его можно переформулировать так: всякое исчисление, порождающее формулы арифметики (—е алгоритм, перечисляющий некоторое множество таких формул) либо *неадекватно* (порождает некоторую ложную формулу), либо *неполно* (не порождает некоторой истинной формулы).

Теперь изложим доказательство теоремы Гёделя. Как мы уже говорили, исчисление — это механизм (алгоритм), который позволяет порождать некоторые формулы языка арифметики (для простоты будем считать, что порождаются только формулы без параметров). Таким образом, возникает некоторое перечислимое множество, которое обычно задают как проекцию разрешимого множества. Именно, вводят некоторое понятие *доказательства*. При этом доказательства являются словами в некотором алфавите. Множество доказательств разрешимо, то есть есть алгоритм, отличающий настоящие доказательства от текстов, который таковыми не являются. Кроме того, есть (также разрешимое) свойство двух слов x и y, которое гласит, что x есть доказательство формулы y. Перенумеровав все доказательства и формулы и выразив указанные разрешимые свойства в языке арифметики, мы приходим к формуле $\mathrm{Proof}(x, y)$, которая истинна, когда x есть номер доказательства формулы с номером y.

Теперь напишем формулу с одним параметром x, которая говорит, что результат подстановки числа x вместо параметра в x-ую формулу с одним параметром не имеет доказательства:

$$\neg\exists z \exists p[\mathrm{Subst}(z, x, x) \wedge \mathrm{Proof}(p, z)]$$

Эта формула имеет единственный параметр (x); пусть её номер в нумерации таких формул равен N. Подставим N вместо параметра. Получится формула без параметров φ. По построению формула φ истинна, когда результат подстановки N в N-ю формулу с одним параметром недоказуем. А этот результат есть сама формула φ, так что она истинна тогда и только тогда, когда недоказуема. Значит, наше исчисление либо позволяет доказать ложную формулу φ (если φ ложна; в таком случае его называют неадекватным), либо не позволяет доказать истинную формулу φ.

Заметим, что доказательства Теорем Геделя и Тарского напоминают классический парадокс лжеца:

УТВЕРЖДЕНИЕ В РАМКЕ ЛОЖНО

Литература

[1] Н. К. Верещагин, А. Шень. *Лекции по математической логике и теории алгоритмов. Часть 1. Начала теории множеств.* М.: МЦНМО, 1999. 128 с.

[2] Н. К. Верещагин, А. Шень. *Лекции по математической логике и теории алгоритмов. Часть 2. Языки и исчисления.* М.: МЦНМО, 2000. 288 с.

[3] Н. К. Верещагин, А. Шень. *Лекции по математической логике и теории алгоритмов. Часть 3. Вычислимые функции.* М.: МЦНМО, 1999. 176 с.

[4] Ю. И. Манин. *Доказуемое и недоказуемое.* М.: Советское радио, 1979. 168 с.

[5] Ю. И. Манин, *Вычислимое и невычислимое.* М.: Советское радио, 1980. 128 с.