Успенский В.А.

Теорема Геделя о неполноте

Москва
«Книга по Требованию»

УДК 51
ББК 22.1
У77

Успенский В.А.
У77 Теорема Геделя о неполноте / Успенский В.А. – М.: Книга по Требованию, 2013. – 109 с.

ISBN 978-5-458-40264-4

Брошюра снабжена шестью приложениями, написанными несколько более сжато, хотя по-прежнему не предполагающими никаких специальных знаний. В первом из них рассматривается вопрос о связи между наличием истинных недоказуемых утверждений и наличием утверждений, не являющихся ни доказуемыми, ни опровержимыми. Во втором доказывается некоторое усиление теоремы Гёделя — теорема Тарского о невыразимости понятия истины. Третье приложение посвящено обоснованию одной из аксиом теории алгоритмов, сформулированных в § 5, а именно, аксиомы арифметичности. С этой целью вводится некоторый конкретный класс алгоритмов — класс адресных программ — и проверяется арифметичность функций, вычисляемых алгоритмами этого класса. В четвертом приложении развитые в § 2 критерии полноты и неполноты применяются к языкам, связанным с так называемыми ассоциативными исчислениями. Пятое приложение посвящено первоначальной формулировке теоремы о неполноте, предложенной самим Гёделем. Шестое приложение содержит упражнения к некоторым из предыдущих разделов. Наконец, последнее приложение содержит ответы и указания к упражнениям. Приложения не зависят друг от друга и могут читаться в любом порядке, за исключением приложения В, отдельные места которого требуют знакомства с введенными в приложении Б понятиями.

ISBN 978-5-458-40264-4

§ 1. ПОСТАНОВКА ЗАДАЧИ

Формулировка теоремы о неполноте, которую мы будем уточнять и доказывать, такова: при определенных условиях

в языке существует недоказуемое истинное утверждение.

В этой формулировке едва ли не каждое слово нуждается в разъяснениях. Сделаем такие разъяснения.

1. **Язык.** Мы не будем давать какое бы то ни было определение языка (поскольку не беремся это сделать с достаточной общностью), а ограничимся теми относящимися к языку понятиями, которые единственно и будут нужны нам для дальнейшего. Таких понятий нам потребуется два: «алфавит языка» и «множество истинных утверждений языка».

1.1. Алфавит. Под *алфавитом* понимается конечный список элементарных (т. е. считающихся не членимыми далее) знаков, называемых *буквами* этого *алфавита*. Конечная цепочка следующих друг за другом букв некоторого алфавита называется *словом* в этом алфавите. Так, слова русского языка (включая и собственные имена) суть слова в 66-буквенном алфавите (33 строчные буквы, 31 прописная буква[1]), дефис, апостроф); десятичные записи натуральных чисел — слова в десятибуквенном алфавите {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. Для называния алфавитов используются обычно прописные русские буквы. Множество всех слов в алфавите Б будем обозначать $Б^{\infty}$. Предполагается, что для каждого языка имеется такой алфавит, что все выражения этого языка (т. е. имена тех или

---

[1]) Кроме твердого и мягкого знаков.

иных предметов, утверждения об этих предметах и т. п.) суть слова в этом алфавите; каждую русскую фразу, например, и даже каждый русский текст можно рассматривать как слово в алфавите, представляющем собой расширение указанного выше 66-буквенного алфавита за счет знаков препинания, знака пробела между словами, знака абзацного отступа (и, быть может, еще некоторых знаков). Предполагая, что выражения языка являются словами в некотором алфавите, мы тем самым налагаем запрет на такое «многоэтажное» выражение, как, например, $\int_a^0 f(x)\,dx$. Этот запрет, однако, не является слишком ограничительным, поскольку все подобные выражения можно при подходящей кодировке «вытянуть в строку». Всякое множество M такое, что $M \subseteq Б^\infty$, называется *словарным в* Б. Просто *словарным* называется множество, словарное в каком-либо алфавите. Сделанное только что предположение может быть теперь сформулировано короче: множество выражений всякого языка словарно.

1.2. Множество истинных утверждений. Предполагается, что в множестве $Б^\infty$, где Б — алфавит рассматриваемого языка, задано подмножество T, называемое множеством «истинных утверждений» (или, короче, просто «истин»). Таким образом, мы опускаем все промежуточные этапы, посредством которых, во-первых, среди слов в алфавите Б выделяются правильно построенные *выражения* языка, получающие определенный смысл при интерпретации (такие, как $2+3$, $x+3$, $x=y$, $x=3$, $2=3$, $2=2$, — в отличие от таких, как $+=x$); во-вторых, среди выражений выделяются так называемые *формулы*, означающие при интерпретации «утверждения, зависящие, быть может, от параметра» (такие, как $x=3$, $x=y$, $2=3$, $2=2$); в-третьих, среди формул выделяются так называемые *замкнутые формулы*, или утверждения, не зависящие от параметра (такие, как $2=3$, $2=2$); и лишь, в-четвертых, среди утверждений выделяются *истинные утверждения* (такие, как $2=2$).

1.3. Для наших целей будет достаточным считать язык полностью заданным, коль скоро задан алфавит Б и подмножество T множества $Б^\infty$. Всякую такую пару $\langle Б, T\rangle$ мы будем называть *фундаментальной парой*.

2. Недоказуемое. «Недоказуемое» значит не являющееся доказуемым, а «доказуемое» значит имеющее доказательство.

3. Доказательство. Хотя термин «доказательство» является едва ли не самым главным в математике [1]), он не имеет точного определения. Понятие доказательства во всей его полноте принадлежит математике не более чем психологии: ведь доказательство — это просто рассуждение, убеждающее нас настолько, что с его помощью мы готовы убеждать других.

3.1. Будучи записанным, доказательство становится словом в некотором алфавите Д (вспомним, что говорилось выше о русских текстах); все доказательства образуют некую (достаточно, впрочем, расплывчатую) совокупность в $Д^*$. Не претендуя на то, чтобы дать точное определение для такого «наивного» или «абсолютного» понятия доказательства (или, что то же самое, для соответствующей совокупности в $Д^*$), мы займемся его формальным аналогом, для которого, однако, сохраним тот же термин «доказательство». Этот аналог в двух существенных чертах будет отличаться от интуитивного понятия [2]): во-первых, мы будем допускать существование разных понятий «доказательства» (что приведет к различным подмножествам в множестве $Д^*$, да и сам алфавит Д может варьироваться); во-вторых, для каждого из таких понятий мы будем требовать наличия эффективного способа, или алгоритма [3]), проверки, является ли данное слово в алфавите Д доказательством или нет. Далее, будем предполагать наличие алгоритма, который по доказательству определяет, доказательством какого утверждения оно является. (В обычных случаях этим утверждением является последнее утверждение в цепочке, образующей доказательство.)

3.2. Итак, окончательное определение таково:

1° Имеются алфавиты Б (*алфавит языка*) и Д (*алфавит доказательств*).

---

[1]) Н. Бурбаки начинает свои «Начала математики» словами «Со времен греков говорить «математика» — значит говорить «доказательство».

[2]) Впрочем, и интуитивное понятие не вовсе лишено этих черт.

[3]) Этот термин уточняется в следующем параграфе.

2° В множестве Д$^\infty$ выделено подмножество D, элементы которого называются *доказательствами*; предполагается наличие алгоритма, позволяющего по произвольному слову в алфавите Д узнавать, принадлежит оно D или нет.

3° Имеется функция δ (*функция выделения доказанного*), у которой область определения Δ удовлетворяет соотношению $D \subseteq \Delta \subseteq$ Д$^\infty$ и которая принимает свои значения в Б$^\infty$; предполагается наличие алгоритма, вычисляющего [1]) эту функцию; доказательство d из D называется доказательством слова $\delta(d)$.

3.3. Тройку $\langle$Д, D, $\delta\rangle$, удовлетворяющую условиям 1°—3°, назовем *дедуктикой* над алфавитом Б.

3.4. Для читателей, знакомых с обычными способами задания понятия «доказательство» посредством «аксиом» и «правил вывода», поясним, почему эти способы могут быть рассмотрены как частный случай определения из п. 3.2. В самом деле, доказательством обычно называют цепочку выражений языка, в которой каждый член или является аксиомой, или получается из предыдущих по одному из правил вывода. Добавляя к алфавиту языка новую букву $*$, мы можем записывать доказательства в виде слов в расширенном таким образом алфавите: цепочка $\langle C_1, \ldots, C_n\rangle$ изображается словом $C_1*C_2* \ldots *C_n$. Функция выделения доказанного выделяет из каждого слова наибольший не содержащий буквы $*$ конец. Требуемые определением из п. 3.2 алгоритмы могут быть легко построены для любого обычно рассматриваемого уточнения понятий «быть аксиомой» и «получаться по одному из правил вывода».

4. Попытки уточнения первоначальной формулировки.

4.1. Первая попытка. При определенных условиях для фундаментальной пары $\langle$Б, $T\rangle$ и дедуктики $\langle$Д, D, $\delta\rangle$ над Б существует слово из T, не имеющее доказательства. Такая формулировка еще слишком неопределенна. К тому же ясно, что можно придумать много дедуктик, в каждой из которых будет очень мало доказуемых слов. В пустой дедуктике (где $D = \varnothing$) вообще нет ни одного доказуемого слова.

---

[1]) Что означают слова «алгоритм вычисляет функцию», будет уточнено в следующем параграфе.

4.2. Вторая попытка. Более естественным является другой подход. Задан некоторый язык в том точном смысле, что задана фундаментальная пара $\langle Б, T\rangle$. Мы теперь ищем дедуктики над Б (содержательно — ищем такие способы доказывания), в которых доказывалось бы как можно больше слов из T, в идеале — все слова из T. Нас интересует ситуация, когда такой дедуктики (в которой каждое слово из T имело бы доказательство) не существует. Итак, нас заинтересовала бы следующая формулировка: при определенных условиях, налагаемых на фундаментальную пару $\langle Б, T\rangle$, не существует дедуктики над Б, в которой каждое слово из T имеет доказательство. Однако пары $\langle Б, T\rangle$ с этим свойством просто не может быть. В самом деле, достаточно положить $Д = Б$, $D = Д^{\infty}$, $\delta(d) = d$ для всякого d из $Д^{\infty}$; тогда всякое слово из $Б^{\infty}$ окажется доказуемым (его доказательством будет оно само).

5. **Непротиворечивость.** Естественно потребовать, чтобы доказуемыми были лишь «истинные утверждения», т. е. слова, принадлежащие множеству T. Назовем дедуктику $\langle Д, D, \delta\rangle$ *непротиворечивой относительно* (или *для*) фундаментальной пары $\langle Б, T\rangle$, коль скоро $\delta(D) \subseteq T$. В дальнейшем будем интересоваться лишь непротиворечивыми дедуктиками. Если имеется язык, то представляется весьма заманчивым найти такую непротиворечивую дедуктику, в которой каждое истинное утверждение было бы доказуемым. Теорема Гёделя в интересующем нас варианте именно и утверждает, что при определенных условиях, налагаемых на фундаментальную пару, этого сделать нельзя.

6. **Полнота.** Назовем дедуктику $\langle Д, D, \delta\rangle$ *полной относительно* (или *для*) фундаментальной пары $\langle Б, T\rangle$, коль скоро $\delta(D) \supseteq T$. Занимающая нас формулировка приобретает такой вид:

при определенных условиях, налагаемых на фундаментальную пару $\langle Б, T\rangle$, *не существует дедуктики над* Б, *полной и непротиворечивой относительно* $\langle Б, T\rangle$.

На этой формулировке мы и остановимся и в следующих параграфах найдем те условия, о которых в ней идет речь.

§ 2. НАЧАЛЬНЫЕ ПОНЯТИЯ ТЕОРИИ АЛГОРИТМОВ И ИХ ПРИМЕНЕНИЯ

Условия, при которых не существует полной непротиворечивой дедуктики, легко формулируются в терминах теории алгоритмов.

Нам достаточно на первых порах лишь самых общих интуитивных представлений об *алгоритме* как о предписании, позволяющем по каждому *исходному данному*, или *аргументу*, из некоторой совокупности *возможных* (для данного алгоритма) *исходных данных* (аргументов) получить *результат* в случае, если таковой существует, или не получить ничего в случае, если для рассматриваемого исходного данного не существует результата [1]). Если для выбранного исходного данного результат существует, говорят, что алгоритм *применим* к этому исходному данному и *перерабатывает* его в этот результат.

Для наших целей будет достаточным — и это позволит избежать лишних обсуждений — считать, что исходные данные и результаты *любого алгоритма* суть слова. Более точно: для каждого алгоритма можно указать некоторый *алфавит исходных данных*, так что все возможные исходные данные являются словами в этом алфавите, и некоторый *алфавит результатов*, так что все результаты являются словами в этом алфавите. Поэтому, чтобы иметь дело с алгоритмами, применяемыми, скажем, к парам слов или к цепочкам слов, мы должны предварительно записать эти образования в виде слов в каком-нибудь алфавите. Для определенности условимся соотносить с каждым алфавитом Б некоторую не входящую в него букву и обозначать эту букву звездочкой (подчеркнем, что, таким образом, эта звездочка в различных ситуациях обозначает различные буквы). Первоначальный алфавит Б, пополненный этой новой буквой, будем обозначать $Б_*$. В п. 3.4 предыдущего параграфа мы уже договорились записывать цепочку $\langle C_1, \ldots, C_n\rangle$ слов в алфавите Б посредством слова $C_1*\ldots*C_n$ в алфавите $Б_*$; в частности, в том же $Б_*$ запишется в виде слова C_1*C_2 и пара $\langle C_1, C_2\rangle$. Пусть, далее, при фиксирован-

[1]) Подчеркнем, что возможные исходные данные — это не те данные, в применении к которым алгоритм дает результат, а те, к которым можно его применять (возможно, безрезультатно).

ном n $Б_1$, $Б_2$, ..., $Б_n$ суть произвольные алфавиты; обозначая по-прежнему через * дополнительную букву, соотнесенную с алфавитом $(Б_1 \cup \ldots \cup Б_n)$ и тем самым заведомо не входящую ни в один из $Б_i$, мы будем отождествлять цепочку $\langle C_1, \ldots, C_n\rangle$, где каждое C_i является словом в $Б_i$, со словом $C_1 * \ldots * C_n$ в алфавите $(Б_1 \cup \ldots \cup Б_n)_*$; совокупность всех таких цепочек (и отождествленных с ними слов) будем обозначать через $Б_1^\infty \times \ldots \times Б_n^\infty$.

Совокупность всех исходных данных, к которым алгоритм применим, называется *областью применимости* алгоритма; каждый алгоритм задает функцию, относящую каждому элементу области применимости соответствующий результат; область определения этой функции совпадает, таким образом, с областью применимости алгоритма; говорят, что рассматриваемый алгоритм *вычисляет* функцию, задаваемую указанным способом. Условимся обозначать через $\boldsymbol{A}(x)$ результат применения алгоритма $\boldsymbol{A}$ к объекту x (при этом $\boldsymbol{A}(\langle x_1, x_2, \ldots, x_n\rangle)$ для краткости будем записывать просто как $\boldsymbol{A}(x_1, \ldots, x_n)$). Тогда определение термина «вычисляет» можно переформулировать следующим образом: алгоритм $\boldsymbol{A}$ вычисляет функцию f, коль скоро $\boldsymbol{A}(x) \simeq f(x)$ для всех x. (Знак $\simeq$ есть знак «условного равенства»; утверждение $A \simeq B$ считается истинным в двух случаях: либо когда выражения A и B оба не определены, либо когда A и B оба определены и обозначают одно и то же.)

Функция, которая вычисляется некоторым алгоритмом, называется *вычислимой*. В части 3° определения понятия доказательства (п. 3.2 предыдущего параграфа) говорится, следовательно, о том, что функция выделения доказанного должна быть вычислимой функцией.

В силу сделанных предположений относительно понятия алгоритма для каждой вычислимой функции можно указать такие два алфавита, что все ее аргументы суть слова в первом из этих алфавитов, а все ее значения — слова во втором из этих алфавитов.

Особый интерес представляют функции, аргументы и значения которых суть натуральные числа[1]). Такие функции условимся называть *числовыми*. Чтобы иметь

[1]) Число 0 мы также считаем натуральным.

право говорить о вычислимых числовых функциях, мы должны ввести в рассмотрение алгоритмы, имеющие дело с числами, а для этого прежде всего необходимо представить числа в виде слов в каком-либо алфавите, называемом в этом случае *цифровым*. Возможны различные способы такого представления, например: 1) двоичная запись чисел в алфавите $\{0, 1\}$; 2) десятичная запись чисел в алфавите $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$; 3) запись чисел в однобуквенном алфавите $\{|\}$, причем число n записывается словом $\underbrace{\| \ldots |}_{n \text{ раз}}$; 4) запись чисел в трехбуквенном алфавите $\{|, (,)\}$, причем число n записывается словом $\underbrace{(\| \ldots |)}_{n \text{ раз}}$, и т. д. Для тех или иных целей выбираются наиболее удобные способы записи. Каждая запись числа (в какой-либо фиксированной системе) называется *цифрой*. Допуская вольность речи, говорят об алгоритмах и вычислимых функциях, оперирующих с числами, имея в виду алгоритмы и вычислимые функции, оперирующие с изображающими эти числа цифрами (в какой-либо выбранной системе записи).

Понятие вычислимой числовой функции, таким образом, выглядит зависящим от принятого способа записи чисел. Однако легко обнаружить, что всякая числовая функция, вычислимая при одной системе записи, будет вычислима и при другой, по крайней мере для широкого класса таких систем. Назовем две системы эквивалентными, если существует алгоритм, дающий по записи произвольного числа в первой системе запись этого же числа во второй системе, а также алгоритм, дающий по записи произвольного числа во второй системе запись этого же числа в первой системе. Приведенные выше примеры систем записи очевидным образом эквивалентны. Покажем, что числовая функция f, вычислимая при одной из двух эквивалентных систем записи, вычислима и при другой системе. Пусть $\boldsymbol{C}$ и $\boldsymbol{D}$ — алгоритмы перехода от первой системы ко второй и обратно, и пусть алгоритм $\boldsymbol{A}$ вычисляет функцию f при первой системе записи (т. е. $\boldsymbol{A}$ вычисляет функцию на цифрах первой системы, индуцированную функцией f). Тогда следующий алгоритм $\boldsymbol{B}$ будет вычислять f при второй системе записи (т. е. вычислять

индуцированную функцию на цифрах второй системы):

$$\boldsymbol{B}(x) \simeq \boldsymbol{CAD}(x).$$

Предписание, задающее алгоритм $\boldsymbol{B}$, может быть словесно выражено следующим образом: «переведи x в первую систему счисления, затем примени алгоритм $\boldsymbol{A}$, полученный результат (если таковой получится) переведи обратно во вторую систему счисления». Аналогичным образом вводимое ниже понятие перечислимого числового множества не зависит от выбора системы записи чисел.

Ввиду сказанного мы позволим себе, коль скоро фиксирована какая-либо система записи чисел, не слишком педантично различать числа и цифры; множество и тех и других будем называть натуральным рядом и обозначать буквой $\mathbb{N}$.

Множество называется *перечислимым*, если оно либо пусто, либо является множеством элементов какой-нибудь вычислимой последовательности (т. е. множеством значений какой-нибудь вычислимой функции, определенной на натуральном ряду); о такой функции (последовательности) говорят, что она *перечисляет* рассматриваемое множество. Очевидно, каждое перечислимое множество словарно.

Пример 1. Множество $\mathbb{N}^2$ всевозможных пар натуральных чисел перечислимо: одна из перечисляющих функций — функция

$$\varphi(n) = \langle a, b\rangle, \quad \text{где} \quad n = 2^a(2b+1) - 1.$$

Пример 2. Множество $Ж^\infty$ всех слов в произвольном алфавите Ж перечислимо. Один из возможных способов построения перечисляющей последовательности таков: упорядочиваем произвольным образом элементы Ж; затем слова в Ж упорядочиваем следующим образом: из слов разной длины предшествующим считается то, которое короче, а на словах одинаковой длины вводим словарный порядок (сравнивая два слова, находим первые слева различающиеся буквы и смотрим, какая из них идет раньше в упорядочении алфавита Ж). Выписывая слова в порядке следования друг за другом, получаем требуемую перечисляющую последовательность. (Может возникнуть вопрос, почему она вычислима, т. е. почему существует алгоритм, дающий по k член a_k этой

последовательности с номером k; искомый алгоритм, например, таков; выписывай члены последовательности, пока их не станет $k+1$; последний из выписанных членов и будет a_k[1]).)

Пример 3. Вычислимая функция f, перечисляющая $Ж^\infty$ и построенная в примере 2, осуществляет взаимно однозначное отображение $\mathbb{N}$ на $Ж^\infty$; поэтому можно говорить об обратной функции f^{-1}, осуществляющей взаимно однозначное отображение $Ж^\infty$ на $\mathbb{N}$. Эта f^{-1} тоже вычислима, поскольку вычисляется следующим алгоритмом: чтобы вычислить $f^{-1}(a)$, вычисляй последовательно $f(0), f(1), f(2), \ldots$ и т. д., пока для некоторого n не получишь $f(n)=a$; это n и есть $f^{-1}(a)$.

Пример 4. Для любых алфавитов $Ж_1$ и $Ж_2$ композиция вычислимых функций, взаимно однозначно отображающих $\mathbb{N}$ на $Ж_2^\infty$ (пример 2) и $Ж_1^\infty$ на $\mathbb{N}$ (пример 3), дает вычислимую же функцию, взаимно однозначно отображающую $Ж_1^\infty$ на $Ж_2^\infty$.

Подмножество S множества A называется *разрешимым* относительно A, коль скоро существует такой алгоритм (*разрешающий S относительно A*), который распознает принадлежность элементов A к S, т. е. такой алгоритм, который все элементы из S перерабатывает в некоторое одно и то же слово x (например, в слово «да»), а все элементы из $A \setminus S$ в некоторое одно и то же, но отличное от x слово y (например, в слово «нет»; разумеется, выбор слов x и y совершенно несуществен). Очевидно, разрешимость множества S относительно A равносильна разрешимости множества $A \setminus S$ относительно того же A. В части 2° определения понятия доказательства требовалось, чтобы множество всех доказательств было разрешимо относительно множества всех слов в алфавите доказательств.

Из определения разрешимости вытекает, что область применимости алгоритма, разрешающего S относительно A, объемлет A. При этом безразлично, что получается в результате применения алгоритма к словам, не лежащим в A. Например, если мы хотим построить алгоритм, отличающий стихи Пушкина от

[1]) Напомним, что последовательность начинается с a_0.