Доказательство теоремы Гёделя о неполноте, основанное на понятиях функционального программирования

А.М. Миронов

Аннотация

В статье излагается новое доказательство теоремы Гёделя о неполноте формальных логических систем, основанное на понятиях функционального программирования.

1 Введение

Теорема Гёделя о неполноте формальных логико-математических систем (называемая ниже просто теоремой Гёделя) занимает центральное положение не только в математической логике, но и во всём современном естествознании. В своей исходной формулировке (см. [1]) теорема Гёделя относится к формальным системам, в которых допускается возможность интерпретации арифметики Пеано (см. [2], [3]), и утверждает, что если формальная система

- является непротиворечивой, и
- обладает возможностями для адекватного выражения в ней утверждений арифметики Пеано

то эта теория не является полной, т.е. не позволяет построить вывод всех истинных утверждений. В этой формулировке её доказательство основано на нумерации формул и доказательств натуральными числами, и имеет высокую сложность.

Однако, если расширить синтаксис формальной арифметики, и рассматривать такие формальные системы, в которых можно выражать свойства символьных строк, то доказательство неполноты таких формальных систем может быть существенно упрощено. Основная идея излагаемого в настоящем тексте доказательства неполноты таких систем заключается в

- моделировании вычислительных процессов, связанных с понятием логического вывода, функциональными программами, и
- естественном представлении описаний и свойств функциональных программ логическими формулами.

2 Аксиоматический метод

Согласно общепринятому мнению, наиболее правильный способ организации научных знаний заключается в представлении их в виде логических следствий, выведенных на основе формальных правил вывода из некоторых исходных утверждений, истинность которых не подвергается сомнению.

Данный способ организации знаний наиболее предпочтителен по следующим причинам.

1. Наличие у некоторой совокупности знаний хорошей логической структуры существенно упрощает овладение этими знаниями.
2. Логическая структуризация знаний облегчает их обработку и существенно повышает её объективность и надёжность, поскольку такая обработка может быть проведена только посредством формальных операций над символьными строками, без привлечения расплывчато–неформальной и субъективной интерпретации понятий, выражаемых этими строками. Кроме того, сведение задачи обработки знаний к задаче выполнения операций над строками обеспечивает возможность автоматизации обработки знаний.
3. Поскольку критерием истинности логически структурированных знаний является их соответствие некоторым синтаксическим правилам, то логическая структуризация знаний облегчает проверку ошибочности в рассуждениях, в которых используются эти знания, поскольку она позволяет свести задачу нахождения ошибок в рассуждениях к задаче проверки правильности использования синтаксических правил при формальных операциях над символьными строками.
4. Представление совокупности знаний в виде логических следствий из некоторых исходных принципов является инструментом синтеза новых знаний, поскольку процесс получения новых знаний может иметь вид формальной комбинации уже установленных утверждений с использованием подходящих правил логического вывода.

Наиболее плодотворные результаты реализация данной точки зрения принесла в математике, где удалось представить все полученнные результаты в виде логических следствий из небольшого числа исходных простых утверждений, называемых **аксиомами**. Метод организации знаний в виде логических следствий из аксиом получил название **аксиоматического метода**.

Аксиоматический метод стал источником бурного развития многих областей математики, и обогатил их новыми плодотворными подходами и глубокими результатами. Наиболее ярко это проявилось в алгебре, которая, благодаря использованию в ней аксиоматического метода, заняла центральное положение в математике.

Некоторое время существовало убеждение, что на базе аксиоматического метода можно построить всю математику, то есть всю совокупность истинных математических утверждений можно представить в виде логических следствий некоторых аксиом, что позволит свести задачу получения новых математических знаний к выполнению формальных операций над символьными строками по заранее заданным правилам.

Однако, как было установлено в 1931 году Гёделем, никакая формальная система (т.е. совокупность аксиом и правил вывода) не является полной, т.е. не может позволить получить таким механическим способом всю совокупность истинных математических утверждений. В частности, утверждение о том, что данная формальная система непротиворечива (т.е. в ней невозможно вывести двух взаимоисключающих предложений) невозможно обосновать методами данной формальной системы.

Теорема Гёделя является убедительным подтверждением тезиса о том, что главный источник развития математики находится за её пределами. Наиболее существенные изменения в математике, приводящие к возникновению новых, более сильных концепций и связанных с ними формальных систем, невозможны в рамках фиксированной системы понятий, они могут произойти только в результате взаимопроникновения и взаимовлияния самых разных областей научной, культурной и практической деятельности. Всякая принципиально новая и более сильная формальная система может быть только изобретением: согласно теореме Гёделя, никаким другим образом её получить невозможно.

3 Строки и функции на них

3.1 Символьные строки

Все объекты, рассматриваемые в математике, можно изобразить символьными строками (которые мы ниже будем называть просто **строками**), и операции на объектах можно представить в виде функций на строках, изображающих эти объекты.

Мы предполагаем, что задано некоторое конечное множество $\mathbf{C}$, элементы которого называются **символами**. В $\mathbf{C}$ входят буквы, цифры, скобки, и некоторые другие символы.

Строкой называется произвольная конечная последовательность символов из $\mathbf{C}$. Совокупность всех строк обозначается символом $\mathbf{S}$. Существует пустая строка, она не содержит ни одного символа.

Все функции на строках, рассматриваемые в настоящей статье, являются частичными, т.е. они могут быть не определены для некоторых значений своих аргументов. Эти функции делятся на два класса:

- базовые функции, и
- функции, определяемые при помощи функциональных программ.

3.2 Базовые функции

Совокупность базовых функций имеет следующий вид.

1. $\texttt{if\_then\_else} : \mathbf{S}^3 \to \mathbf{S}$
   Для каждой тройки $(u, v, w) \in \mathbf{S}^3$

   $$\texttt{if\_then\_else}(u, v, w) \stackrel{\text{def}}{=} \begin{cases} v, & \text{если } u \text{ состоит из} \\ & \text{одного символа 1} \\ w, & \text{иначе} \end{cases}$$

   Ниже знакосочетание $\texttt{if\_then\_else}(u, v, w)$ будет сокращённо записываться в виде

   $$u \ ? \ v : w$$

2. $\texttt{head} : \mathbf{S} \to \mathbf{S}$
   Эта функция определена только на непустых строках, она сопоставляет каждой непустой строке u строку, состоящую из первого символа строки u.

3. $\texttt{tail} : \mathbf{S} \to \mathbf{S}$
   Эта функция тоже определена только на непустых строках. Она сопоставляет каждой непустой строке u строку, получаемую из u удалением первого символа.

4. $\texttt{conc} : \mathbf{S}^2 \to \mathbf{S}$
   Для каждой пары $(u, v) \in \mathbf{S}^2$ строка $\texttt{conc}(u, v)$ представляет собой конкатенацию строк u и v, т.е.

   - строку $a_1 \ldots a_n \, b_1 \ldots b_m$, если u и v имеют вид соответственно $a_1 \ldots a_n$ и $b_1 \ldots b_m$, где $m, n > 0$, и
   - строку v, если строка u пуста,
   - строку u, если строка v пуста.

5. $\texttt{eq} : \mathbf{S}^2 \to \mathbf{S}$
   Для каждой пары $(u, v) \in \mathbf{S}^2$

   $$\texttt{eq}(u, v) \stackrel{\text{def}}{=} \begin{cases} 1, & \text{если } u = v \\ 0, & \text{иначе} \end{cases}$$

3.3 Структурированные строки

Некоторые строки мы будем называть **структурированными**. Понятие структурированной строки (СС) определяется индуктивно. При определении понятия СС используется вспомогательное понятие **атома**.

- **Атомом** называется произвольная строка, не содержащая фигурных скобок (т.е. символов { и }).

  Каждый атом является СС.

- Если строки $u_1, \ldots, u_n$ (где $n \geq 1$) являются СС, то строка

$$\{u_1 \ldots u_n\}$$

(получаемая приписыванием к конкатенации строк $u_1, \ldots, u_n$ скобки { слева и скобки } справа) является СС.

3.4 Переменные, константы, выражения и подстановки

СС вида $\{\{\texttt{var}\}u\}$ (где var – строка из трёх символов v, a и r) мы будем называть **переменными**.

СС вида $\{\{\texttt{con}\}u\}$ мы будем называть **константами**. Мы будем обозначать константу вида $\{\{\texttt{con}\}u\}$ знакосочетанием $\underline{u}$. Строка u называется **интерпретацией** константы $\underline{u}$.

СС вида $\{\{\texttt{fun}\}\{n\}u\}$ (где n – атом, представляющий собой запись некоторого положительного целого числа) мы будем называть **функциональными символами (ФС)**.

Если ФС f имеет вид $\{\{\texttt{fun}\}\{n\}u\}$, то число n в записи этого ФС называется **арностью** ФС f, и обозначается через $ar(f)$. Оно равну количеству аргументов у функции, соответствующей этому ФС.

Совокупность **базовых ФС** состоит из следующих пяти ФС:

1. $\{\{\texttt{fun}\}\{3\}\{\texttt{if\_then\_else}\}\}$
2. $\{\{\texttt{fun}\}\{1\}\{\texttt{head}\}\}$
3. $\{\{\texttt{fun}\}\{1\}\{\texttt{tail}\}\}$
4. $\{\{\texttt{fun}\}\{2\}\{\texttt{conc}\}\}$
5. $\{\{\texttt{fun}\}\{2\}\{\texttt{eq}\}\}$

Выражения строятся из переменных, констант и ФС:

1. все переменные и константы являются выражениями, и
2. для каждого ФС f, и каждого кортежа выражений

$$e_1, \ldots, e_n \quad (\text{где } n = ar(f))$$

СС

$$\{\{\texttt{expr}\}f e_1 \ldots e_n\} \tag{1}$$

является выражением.

В целях удобства чтения, выражение вида (1) мы будем записывать знакосочетанием

$$f(e_1, \ldots, e_n)$$

Подстановка – это СС вида

$$\{\{\texttt{subst}\}\{x_1 \ldots x_n\}\{e_1 \ldots e_n\}\} \tag{2}$$

где

- $x_1, \ldots, x_n$ – список различных переменных, и
- $e_1, \ldots, e_n$ – список выражений.

Для каждой подстановки θ вида (2) и каждого выражения e знакосочетание θe обозначает выражение, получаемое из e заменой для каждого $i \in \{1, \ldots, n\}$ каждого вхождения переменной x_i в e на выражение e_i.

В целях удобства чтения, выражение вида

$$\{\{\texttt{subst}\}\{x_1 \ldots x_n\}\{e_1 \ldots e_n\}\}e$$

мы будем записывать знакосочетанием

$$e(e_1/x_1, \ldots, e_n/x_n)$$

3.5 Функциональные программы

Функциональное уравнение - это СС вида

$$\{\{\texttt{fun\_equation}\}f\{x_1 \ldots x_n\}e\} \tag{3}$$

где

- f – небазовый ФС,
- $x_1, \ldots, x_n$ – список различных переменных, где $n = ar(f)$ и
- e – выражение, такое, что каждая переменная, входящая в e, содержится в множестве $\{x_1, \ldots, x_n\}$.

В целях удобства чтения, функциональное уравнение вида (3) мы будем записывать знакосочетанием

$$f(x_1, \ldots, x_n) = e$$

Функциональная программа (ФП) – это СС, представляющая собой список функциональных уравнений

$$\left\{ \begin{array}{l} f_1(x_{11}, \ldots, x_{1k_1}) = e_1 \\ \ldots \\ f_n(x_{n1}, \ldots, x_{nk_n}) = e_n \end{array} \right\} \tag{4}$$

где

- $f_1, \ldots, f_n$ – различные небазовые ФС, и
- каждый небазовый ФС, входящий в какое-либо из выражений $e_1, \ldots, e_n$, содержится в списке

$$f_1, \ldots, f_n \tag{5}$$

ФП (4) является определением функций, соответствующих ФС $f_1, \ldots, f_n$. Мы будем предполагать, что аргументами и значениями всех рассматриваемых функций являются СС. Для каждого $i = 1, \ldots, n$ мы будем обозначать функцию, соответствующую ФС f_i, тем же символом f_i. Функции $f_1, \ldots, f_n$, вычисляются стандартной рекурсией: если требуется вычислить значение функции f_i на кортеже аргументов $(u_1, \ldots, u_{k_i})$, то для этого вычисляется значение выражения

$$e_i\,(\,\underline{u_1}/x_{i1}, \ldots, \underline{u_{k_i}}/x_{ik_i}) \tag{6}$$

и искомое значению $f_i(u_1, \ldots, u_{k_i})$ по определению полагается равным значению выражения (6).

Вычисление значения выражения (6) происходит по следующему рекурсивному правилу.

- Если (6) имеет вид $\underline{u}$, то его значением является u.
- Если выражение (6) имеет вид

  $$f(e'_1, \ldots, e'_m)$$

  где f – базовый ФС, отличный от `if_then_else`, то вычисляются значения выражений $e'_1, \ldots, e'_m$, и значение выражения (6) полагается равным значению функции f на списке значений выражений $e'_1, \ldots, e'_m$.
- Если выражение (6) имеет вид

  $$e'_1 \ ? \ e'_2 : e'_3$$

  то сначала вычисляется значение выражения e'_1, и
  - если оно равно строке из одного символа 1, то вычисляется значение выражения e'_2,
  - иначе - вычисляется значение выражения e'_3

  и значение выражения (6) полагается равным значению e'_2 или e'_3 соответственно.
- Если выражение (6) имеет вид

  $$f_j(e'_1, \ldots, e'_{k_j})$$

  где ФС f_j принадлежит списку (5), то значение выражения (6) по определению равно значению выражения

  $$e_j(e'_1/x_{j1}, \ldots, e'_{k_j}/x_{jk_j})$$

  которое вычисляется по тому же правилу, по которому вычисляется значение выражения (6).

Если описанный выше процесс не завершается, то мы будем считать, что значение функции f_i на кортеже аргументов $(u_1, \ldots, u_{k_i})$ не определено.

3.6 Примеры ФП

Приведём три примера ФП. Ниже

- символ ε обозначает константу $\{\{\mathtt{con}\}\}$ (т.е. интерпретацией константы ε является пустая строка), и
- знакосочетания *eq*, *conc*, *head* и *tail* обозначают соответствующие базовые ФС.

1. ФП, определяющая функцию *rev*, которая преобразует каждую строку в строку, записанную в обратном порядке (т.е. $rev(a_1 \ldots a_n) = a_n \ldots a_1$):

   $$\begin{array}{lll} rev(x) = eq(x, \varepsilon) & ? & \varepsilon \\ & : & conc(rev(tail(x)), head(x)) \end{array}$$

2. В следующем примере мы предполагаем, что множество символов **C** линейно упорядочено, и имеется дополнительная базовая функция *leq* от двух аргументов, которая
   - определена только на строках из одного символа, и
   - принимает на паре своих аргументов значение
     - 1, если первый аргумент не превосходит второго, и
     - 0 - в противном случае.

   Ниже представлена ФП, определяющая функцию *sort*. Данная функция преобразует каждую строку x в строку $sort(x)$, обладающую следующими свойствами:
   - компоненты строки $sort(x)$ образуют неубывающую последовательность, и
   - каждый символ входит в строку $sort(x)$ столько же раз, сколько раз он входит в x

   т.е. функция *sort* осуществляет сортировку своего аргумента.

   $$\left\{ \begin{array}{l} sort(x) = \quad eq(x, \varepsilon) \ ? \ \varepsilon \\ \qquad\qquad : insert(head(x), sort(tail(x))) \\ insert(a, y) = eq(y, \varepsilon) \ ? \ a \\ \quad : \left(\begin{array}{l} leq(a, head(y)) \ ? \ conc(a, y) \\ : conc(head(y), insert(a, tail(y)) \end{array} \right) \end{array} \right.$$

3. Поиск заданной подстроки x в строке y: функция *substring* возвращает
   - 1, если её первый аргумент является подстрокой второго аргумента, и
   - 0, иначе.

   $$\left\{ \begin{array}{l} substring(x, y) = (prefix(x, y)) \ ? \ \underline{1} \\ \quad : \left(\begin{array}{l} eq(y, \varepsilon) \ ? \ \underline{0} \\ \quad : substring(x, tail(y)) \end{array} \right) \\ prefix(x, y) = eq(x, \varepsilon) \ ? \ \underline{1} \\ \quad : eq(y, \varepsilon) \ ? \ \underline{0} \\ \qquad : \left(\begin{array}{l} (head(x) = head(y)) \ ? \\ \quad prefix(tail(x), tail(y)) \\ \quad : \underline{0} \end{array} \right) \end{array} \right.$$

3.7 Свойство алгоритмической полноты ФП

Множество всех ФП обладает свойством **алгоритмической полноты**, которое заключается в следующем: каждая частичная функция на строках, вычислимая в некотором интуитивном смысле, может быть определена при помощи некоторой ФП. Это свойство доказывается посредством моделирования функциональными программами машин Тьюринга. Доказательство основано на предположении, что множество машин Тьюринга обладает свойством алгоритмической полноты.

Пусть имеется некоторая машина Тьюринга с множеством состояний Q и алфавитом A символов, которые могут быть написаны в ячейках ленты. Мы будем предполагать, что A не содержит фигурных скобок (если такие скобки присутствуют в A, то мы их как-нибудь переобозначим). Также мы будем предполагать, что

- введённое выше множество символов $\mathbf{C}$ содержит каждый символ, входящий в A, и
- A содержит **пробельный символ**, который мы будем обозначать через β.

Пусть работа этой машины описывается отображением

$$\delta : Q \times A \to Q \times A \times \{\mathtt{left}, \mathtt{right}\}$$

Тогда ФП, соответствующая этой машине, определяется следующим образом. Сопоставим каждому состоянию $q \in Q$ некоторый небазовый ФС арности 2, который мы будем обозначать так же, как и это состояние, т.е. через q. Искомая ФП состоит из следующих уравнений:

- уравнение, соответствующее функции f, которую вычисляеет эта машина, имеет вид

  $$f(x) = q_0(\varepsilon, x)$$

  где q_0 – начальное состояние машины
- для каждого $q \in Q$ ФП содержит уравнение вида $q(x, y) = \ldots$, где
  - переменная x соответствует записи на ленте машины слева от головки, прочитанной справа налево, и
  - переменная y соответствует записи на ленте под головкой и справа от неё.

  Если
  - множество A имеет вид $\{a_1, \ldots\}$,
  - q – некоторое незаключительное состояние, и
  - $\delta(q, a_1)$ имеет вид, например, $(q_i, a_j, \mathtt{right})$

  то уравнение, соответствующее состоянию q, имеет вид

  $$\begin{array}{l} q(x,y) = eq(y,\varepsilon)\ ?\ q(x,\underline{\beta}) \\ \quad : eq(head(y),\underline{a_1})\ ?\ q_i(conc(\underline{a_j},x),tail(y)) \\ \quad\quad : eq(head(y),\underline{a_2})\ ?\ldots \end{array}$$

  Если q – заключительное состояние, то соответствующее ему уравнение имеет вид

  $$q(x, y) = remove\_blanks(y)$$

  где функция, соответствующая ФС $remove\_blanks$, действует на свой аргумент путём удаления пробельных символов в его правой части, т.е.

  $$\left\{\begin{array}{l} remove\_blanks(x) = eq(x,\varepsilon)\ ?\ \varepsilon \\ \quad : eq(last(x),\beta)\ ?\ remove\_blanks(rem\_last(x)) \\ \quad\quad : x \\ last(x) = eq(tail(x),\varepsilon)\ ?\ head(x) \\ \quad : last(tail(x)) \\ rem\_last(x) = eq(tail(x),\varepsilon)\ ?\ \varepsilon \\ \quad : conc(head(x), rem\_last(tail(x))) \end{array}\right.$$

3.8 Интерпретируемые ФС

Пусть u – ФП вида (4). Тогда для каждого $i = 1, \ldots, n$ ФС вида

$$\{\{\mathtt{fun}\}\{k_i\}\{\mathtt{interpreted}\}u\{i\}\} \tag{7}$$

называется **интерпретируемым** ФС, и ему соответствует та же самая функция

$$f_i : \mathbf{S}^{k_i} \to \mathbf{S}$$

которая соответствует ФС f_i в ФП (4).

В целях удобства чтения, мы будем обозначать ФС вида (7) знакосочетанием f_i^u.

4 Формулы

4.1 Понятие формулы

Выражение e называется **интерпретируемым**, если это либо переменная, либо константа, либо имеет вид

$$f(e_1, \ldots, e_n)$$

где f – базовый или интерпретируемый ФС.

Для каждой пары e_1, e_2 интерпретируемых выражений СС

$$\{\{\mathtt{elem\_fm}\}\ e_1 = e_2\} \tag{8}$$

является формулой. Формулы вида (8) называются **элементарными формулами**. В целях удобства чтения, формулу вида (8) мы будем записывать знакосочетанием

$$e_1 = e_2$$

Остальные формулы строятся стандартным образом из элементарных формул при помощи булевских связок и кванторов:

- если A и B - формулы, то СС

$$\begin{array}{l}\{\{\mathtt{fm}\}\ A\ \wedge B\},\\ \{\{\mathtt{fm}\}\ A\ \vee B\},\\ \{\{\mathtt{fm}\}\ A\ \rightarrow B\},\\ \{\{\mathtt{fm}\}\ A\ \leftrightarrow B\},\\ \{\{\mathtt{fm}\}\ \neg A\}\end{array} \tag{9}$$

являются формулами, и

- если A – формула и x – переменная, то СС

$$\{\{\mathtt{fm}\}\ \forall\ x\ A\}\ \text{и}\ \{\{\mathtt{fm}\}\ \exists\ x\ A\}$$

являются формулами.

Мы будем обозначать неэлементарные формулы так, как это принято в математической логике, т.е. знакосочетаниями

$$A \wedge B, A \vee B, \ldots$$

соответственно.

Понятия свободного и связанного вхождения переменных в формулы определяются стандартным образом (см., например, [2]).

Если A – формула, x – переменная, и e – выражение, то знакосочетание $A(e/x)$ обозначает формулу, получаемую из A заменой каждого свободного вхождения переменной x в A на выражение e. При этом, если необходимо, производятся переименования связанных переменных в A (подробности см. в [2], гл. 2).

4.2 Истинность формул

Пусть A – некоторая формула, и $x_1, \ldots, x_n$ – список всех переменных, имеющих свободные вхождения в A.

Означиванием свободных переменных в формуле A называется произвольный список ξ вида

$$a_1, \ldots, a_n$$

каждая компонента a_i которого является СС и соответствует переменной x_i (с тем же номером) из списка $x_1, \ldots, x_n$.

Понятие **истинности** формулы A на означивании ξ определяется индукцией по построению формулы A:

- если A имеет вид $e_1 = e_2$, то она является истинной на ξ, если значения выражений

$$e_1(\underline{a_1}/x_1, \ldots, \underline{a_n}/x_n) \quad \text{и} \quad e_2(\underline{a_1}/x_1, \ldots, \underline{a_n}/x_n)$$

либо оба неопределены, либо оба определены и совпадают,

- истинность неэлементарных формул на означиваниях определяется стандартным образом.

Формула называется **истинной**, если она истинна на произвольном означивании.

5 Формальные системы

5.1 Понятие формальной системы

Формальная система представляет собой совокупность аксиом и правил вывода, и будет ниже обозначаться символом $\mathcal{T}$. В излагаемых ниже формулировках символы A, B обозначают произвольные формулы, символы e, e_1, e_2 - произвольные выражения, символы x, y, z – произвольные переменные.

5.1.1 Аксиомы

Аксиома - это формула, принадлежащая одному из следующих шести множеств.

1. Пусть

   - T – произвольная тавтология логики высказываний, и
   - $p_1, \ldots, p_n$ - список всех булевых переменных, входящих в T.

   Тогда для каждого списка формул вида $A_1, \ldots, A_n$ формула

   $$T(A_1/p_1, \ldots, A_n/p_n)$$

   (получаемая из T заменой для каждого $i = 1, \ldots, n$ каждого вхождения переменной p_i в T на соответствующую ей формулу A_i) является аксиомой.

2. Кванторные аксиомы:

$$\neg(\forall x A) \leftrightarrow \exists x(\neg A), \quad \neg(\exists x A) \leftrightarrow \forall x(\neg A) \tag{10}$$

$$\begin{array}{l}A \leftrightarrow \forall x A,\ A \leftrightarrow \exists x A\\ (\text{где } x \text{ не имеет своб. вхождений в } A)\end{array} \tag{11}$$

$$A(e/x) \rightarrow \exists x A \tag{12}$$

3. Аксиомы равенства

$$x = x \tag{13}$$

$$(x = y) \rightarrow (y = x) \tag{14}$$

$$(x = y) \rightarrow ((y = z) \rightarrow (x = z)) \tag{15}$$

$$(e_1 = e_2) \rightarrow (e(e_1/x) = e(e_2/x)) \tag{16}$$

$$(e_1 = e_2) \rightarrow (A(e_1/x) \leftrightarrow A(e_2/x)) \tag{17}$$

4. Аксиомы для базовых функций на строках

   - $(x = \underline{1}) \rightarrow ((x\ ?\ y : z) = y)$
   - $(x = \underline{0}) \rightarrow ((x\ ?\ y : z) = z)$
   - $(eq(x, y) = \underline{1}) \leftrightarrow (x = y)$
   - $(eq(x, y) = \underline{0}) \leftrightarrow \neg(x = y)$
   - $\neg(x = \varepsilon) \rightarrow (conc(head(x), tail(x)) = x)$
   - для
     - каждого базового ФС f, и

– каждого списка СС $a_1, \ldots, a_n, b$, такого, что значение функции f на кортеже аргументов $(a_1, \ldots, a_n)$ равно b

формула

$$f(\underline{a_1}, \ldots, \underline{a_n}) = \underline{b}$$

является аксиомой

5. Пусть u – ФП вида (4), и i – произвольный индекс из множества $\{1, \ldots, n\}$. Тогда формула

$$f_i^u(x_{i1}, \ldots, x_{ik_i}) = e_i^u$$

где e_i^u получается из выражения e_i в правой части i–го функционального уравнения в этой ФП заменой каждого ФС f_j на соответствующий ему интерпретируемый ФС f_j^u, является аксиомой.

6. Дополнительное множество аксиом Add_Ax, обладающее следующими свойствами:

   - каждая формула из Add_Ax истинна
   - множество Add_Ax **разрешимо**, т.е. существует алгоритм, который для каждой формулы определяет, принадлежит ли она этому множеству, или нет.

Из данного определения следует, что множество всех аксиом обладает теми же свойствами, что и множество Add_Ax, т.е.

- каждая аксиома является истинной формулой, и
- множество всех аксиом разрешимо.

Разрешимость множества аксиом следует из того, что принадлежность формулы к любой из вышеперечисленных групп (кроме последней) определяется по синтаксической структуре данной формулы.

Поскольку мы не детализируем вид аксиом из последней группы, то в действительности упомянутый выше символ $\mathcal{T}$ обозначает не конкретную формальную систему, а произвольную формальную систему, множество аксиом которой может быть представлено в виде совокупности вышеперечисленных групп.

5.1.2 Правила вывода

Правила вывода позволяют получать из одних формул другие формулы, и имеют следующий вид:

1. modus ponens:

$$\frac{A \to B, \quad A}{B}$$

2. навешивание кванторов

$$\frac{A \to B}{\forall x A \to \forall x B} \qquad \frac{A \to B}{\exists x A \to \exists x B} \tag{18}$$

В каждом правиле вывода над чертой изображены одна или две формулы (называемые **посылками**), из которых выводится формула, расположенная под чертой (называемая **заключением**). Нетрудно доказать, что если посылки (или посылка) какого-либо из этих правил истинны, то заключение тоже будет истинным.

5.1.3 Понятие доказательства

Пусть A – некоторая формула. **Доказательством** (в системе $\mathcal{T}$) формулы A называется СС вида

$$\{\{\texttt{proof}\} A_1 \ldots A_n\} \tag{19}$$

где $A_1, \ldots, A_n$ – формулы, причём

- $A_n = A$, и
- для каждого $i = 1, \ldots, n$ формула A_i является
  - либо аксиомой системы $\mathcal{T}$,
  - либо заключением некоторого правила вывода, посылки которого содержатся в множестве $\{A_1, \ldots, A_{i-1}\}$.

Формула называется **доказуемой** (в $\mathcal{T}$), если существует доказательство в $\mathcal{T}$ этой формулы.

Из вышесказанного следует, что если формула является доказуемой, то она является истинной.

Из свойства алгоритмической полноты ФП вытекает существование ФП *prf*, определяющей функцию с двумя аргументами, которая на паре аргументов (u, v) принимает значение

- 1, если u является доказательством v (в $\mathcal{T}$), и
- 0, иначе.

5.2 Интерпретируемость в $\mathcal{T}$ арифметики Пеано

Описанная выше система $\mathcal{T}$ является более общей, чем арифметика Пеано (аксиомы и правила вывода которой см. в [2], гл. 3), в том смысле, что объекты и формулы арифметики Пеано можно интерпретировать в языке системы $\mathcal{T}$:

- натуральные числа можно интерпретировать строками из единиц: каждое положительное число n интерпретируется строкой из n единиц, число 0 интерпретируется пустой строкой,
- функция сложения интерпретируется базовой функцией *conc* конкатенации строк
- функция взятия следующего числа интерпретируется функцией *succ*, определяемой следующей ФП:

$$succ(x) = conc(x, \underline{1})$$

- функция умножения интерпретируется функцией *mult*, определяемой следующей ФП:

$$mult(x, y) = eq(x, \varepsilon)\ ?\ \varepsilon : conc(x, mult(tail(x), y))$$

Для каждой формулы A языка арифметики Пеано обозначим символом $I(A)$ формулу в языке системы $\mathcal{T}$, которая получается из A

- заменой констант и функциональных символов на соответствующие им константы и функциональные символы системы $\mathcal{T}$, и
- заменой каждой подформулы вида $\forall x B$ на формулу вида

$$\forall x((nat(x) = \underline{1}) \to B)$$

где функция nat принимает на своём аргументе значение

  - 1, если он является строкой из единиц или пустой строкой, и
  - 0 - иначе.

Нетрудно проверить, что для каждой аксиомы A арифметики Пеано формула $I(A)$ является истинной.

Если система $\mathcal{T}$ такова, что её последняя группа аксиом содержит формулы вида $I(A)$, где A – какая-либо из аксиом арифметики Пеано, то нетрудно установить, что для каждой формулы A, доказуемой в арифметике Пеано, формула $I(A)$ будет доказуема в $\mathcal{T}$.

5.3 Вспомогательные утверждения

В этом пункте мы докажем две леммы, необходимые для доказательства теоремы Гёделя.

Лемма 1.
Пусть

- f – базовый или интерпретируемый ФС, и
- $a_1, \ldots, a_k, b$ – список СС, такой, что формула

$$f(\underline{a_1}, \ldots, \underline{a_k}) = \underline{b} \tag{20}$$

является истинной.

Тогда формула (20) доказуема.

Доказательство.
Если f – базовый ФС, то формула (20) является аксиомой (из четвертой группы).

Если же f – интерпретируемый ФС, т.е. f имеет вид f_i^u, где

- u – ФП, которая получается из ФП вида (4) заменой каждого небазового ФС f_j на ФС f_j^u, и
- $i \in \{1, \ldots, n\}$

то из описания процесса вычисления значения выражения в пункте 3.5 следует, что, существует последовательность выражений

$$e_1, \ldots, e_m \tag{21}$$

такая, что

- $e_1 = f_i^u(\underline{a_1}, \ldots, \underline{a_k})$
- $e_m = \underline{b}$
- для каждого $l = 1, \ldots, m-1$ выражение e_{l+1} получается из выражения e_l заменой одного из его подвыражений t на выражение s, причём имеет место одно из следующих условий:

  1. – t имеет вид $f(\underline{u_1}, \ldots, \underline{u_n})$, где f – базовый ФС, отличный от `if_then_else`,
     – s равно константе $\underline{v}$, где v – значение базовой функции f на кортеже аргументов $(u_1, \ldots, u_n)$
  2. – t имеет вид $\underline{1}\ ?\ e'_1 : e'_2$
     – s равно e'_1
  3. – t имеет вид $\underline{0}\ ?\ e'_1 : e'_2$
     – s равно e'_2
  4. – t имеет вид $f_j^u(e'_1, \ldots, e'_{k_j})$
     – s равно $e_j^u(e'_1/x_{j1}, \ldots, e'_{k_j}/x_{jk_j})$

  Заметим, что в каждом из этих случаев формула $t = s$ является доказуемой.

Для каждого $l = 1, \ldots, m-1$ связь между выражениями e_l и e_{l+1} в списке (21) можно выразить следующим образом: существует выражение e, такое, что

$$e_l = e(t/x), \quad e_{l+1} = e(s/x)$$

где формула $t = s$ доказуема. Используя аксиомы вида (16) и правило modus ponens, заключаем, что для каждого $j = 1, \ldots, m-1$ формула $e_j = e_{j+1}$ доказуема.

На основании аксиом транзитивности равенства, отсюда следует доказуемость формулы $e_1 = e_m$, которая совпадает с (20). ■

Лемма 2 (лемма о неподвижной точке).
Для каждой формулы A с одной свободной переменной x существует замкнутая (т.е. не содержащая свободных переменных) формула φ, такая, что доказуема формула

$$\varphi \leftrightarrow A(\underline{\varphi}/x) \tag{22}$$

Доказательство.
Мы можем предполагать, что все вхождения x в A являются свободными (если это не так, то переименуем все связанные вхождения x в A).

Обозначим знакосочетанием $subst$ интерпретируемый ФС, которому соответствует функция

$$subst : \mathbf{S}^2 \to \mathbf{S}$$

со следующим свойством: для каждой пары СС (a, b) $subst(a, b)$ получается из a заменой в ней каждого вхождения переменной x на СС b.

Обозначим символом B формулу $A(subst(x, x)/x)$.

Искомая формула φ имеет вид $B(\underline{B}/x)$. Поскольку

- единственной свободной переменной в B является x, и
- все вхождения x в B при подстановке $(\underline{B}/x)$ заменяются на константу $\underline{B}$

то φ замкнута.

Поскольку переход от B к φ заключается в замене в СС B каждого вхождения переменной x на СС $\underline{B}$, то.переход от $\underline{B}$ к $\underline{\varphi}$ заключается в замене в СС $\underline{B}$ каждого вхождения переменной x на СС $\underline{B}$, т.е. истинна формула

$$subst(\underline{B}, \underline{B}) = \underline{\varphi} \qquad (23)$$

Согласно лемме 1, формула (23) доказуема. Используя аксиому вида (17) и правило modus ponens, заключаем, что из доказуемости формулы (23) следует доказуемость формулы

$$A(subst(\underline{B}, \underline{B})/x) \leftrightarrow A(\underline{\varphi}/x)$$

которая совпадает с (22), так как левая часть в ней совпадает с φ. ■

6 Теорема Гёделя

Теорема Гёделя утверждает, что если формальная система $\mathcal{T}$ непротиворечива (т.е. в ней недоказуема ложная формула), то существует замкнутая формула φ, такая, что обе формулы φ и $\neg\varphi$ недоказуемы в $\mathcal{T}$. Поскольку одна из формул φ, $\neg\varphi$ истинна, то, следовательно, существует истинная формула, которая не является доказуемой в системе $\mathcal{T}$.

Для построения такой формулы рассмотрим формулу

$$\neg\, \exists y\, (prf(y, x) = \underline{1})$$

Единственной свободной переменной этой формулы является x. По лемме о неподвижной точке, существует замкнутая формула φ, такая, что доказуема формула

$$\varphi \;\leftrightarrow\; \neg\, \exists y\, (prf(y, \underline{\varphi}) = \underline{1}) \qquad (24)$$

Ниже мы докажем, что обе формулы φ и $\neg\varphi$ недоказуемы.

Мы будем использовать следующее обозначение: для каждой формулы A знакосочетание $\Box A$ является сокращённой записью формулы

$$\exists x\, (prf(x, \underline{A}) = \underline{1})$$

Формула $\Box A$ выражает утверждение о том, что формула A является доказуемой (в системе $\mathcal{T}$). Используя данное обозначение, можно переписать (24) в виде

$$\varphi \;\leftrightarrow\; \neg\, \Box\varphi \qquad (25)$$

Лемма 3.

Если формула A доказуема, то доказуема формула $\Box A$.

Доказательство.

Если формула A доказуема, то существует СС u, такая, что формула

$$prf(\underline{u}, \underline{A}) = \underline{1} \qquad (26)$$

является истинной. Согласно лемме 1, формула (26) доказуема. По правилу modus ponens, из (26) и из формулы

$$(prf(\underline{u}, \underline{A}) = \underline{1}) \rightarrow \exists x\, (prf(x, \underline{A}) = \underline{1}) \qquad (27)$$

(которая является аксиомой вида (12)) получаем доказуемость заключения импликации (27), которое совпадает с $\Box A$. ■

Теперь можно приступить к доказательству того, что формулы φ и $\neg\varphi$ недоказуемы.

1. Если доказуема φ, то по лемме 3 доказуема $\Box\varphi$. Используя доказуемую формулу (25), которую можно переписать эквивалентным образом в виде

   $$\neg\varphi \;\leftrightarrow\; \Box\varphi \qquad (28)$$

   получаем доказуемость $\neg\varphi$. Это невозможно, если $\mathcal{T}$ непротиворечива.

2. Если доказуема $\neg\varphi$, то из доказуемости (28) следует, что доказуема $\Box\varphi$, т.е. доказуема формула

   $$\exists x\, (prf(x, \underline{\varphi}) = \underline{1})$$

   Поскольку каждая доказуемая формула является истинной, то, следовательно, существует СС u, такая, что формула

   $$prf(\underline{u}, \underline{\varphi}) = \underline{1}$$

   является истинной, т.е. существует СС u, являющаяся доказательством формулы φ, т.е. формула φ доказуема, что невозможно, если $\mathcal{T}$ непротиворечива. ■

Отметим, что недоказуемость некоторой истинной формулы невозможно устранить путём расширения системы $\mathcal{T}$ добавлением ней произвольного перечислимого множества дополнительных аксиом, т.е. такого множества дополнительных аксиом, каждая из которых является истинной формулой и имеет вид $f(u)$, где

- f – некоторая фиксированная всюду определённая вычислимая функция на СС, и
- u - произвольная СС

потому, что получившаяся система $\mathcal{T}'$ будет иметь то же множество доказуемых формул, что и некоторая система $\mathcal{T}''$ с разрешимым множеством дополнительных аксиом, а именно, $\mathcal{T}''$ получается из $\mathcal{T}$ добавлением аксиом вида

$$f(u) \wedge (u = u)$$

Система $\mathcal{T}''$ будет удовлетворять тем же условиям, что и исходная система $\mathcal{T}$. Поэтому, если она непротиворечива, то в ней тоже будет недоказуема некоторая истинная формула.

7 Недоказуемость непротиворечивости

Обозначим знакосочетанием $Consis(\mathcal{T})$ формулу $\neg\,\Box\mathbf{0}$, где $\mathbf{0}$ – тождественно ложная формула (отрицание тавтологии). Формула $Consis(\mathcal{T})$ выражает свойство непротиворечивости системы $\mathcal{T}$.

Можно доказать, что доказуема эквиваленция

$$\varphi \leftrightarrow Consis(\mathcal{T}) \tag{29}$$

где φ – формула, определённая в предыдущем пункте. Как было установлено, если $\mathcal{T}$ непротиворечива, то формула φ недоказуема, поэтому формула $Consis(\mathcal{T})$ также недоказуема (в системе $\mathcal{T}$).

Утверждение о недоказуемости непротиворечивости формальной системы средствами самой этой системы обычно называют **второй теоремой Гёделя**.

Опишем схему доказательства эквиваленции (29). Из определений φ и $Consis(\mathcal{T})$ следует, что доказуемость (29) равносильна доказуемости эквиваленции

$$\Box\varphi \leftrightarrow \Box\mathbf{0} \tag{30}$$

Можно доказать, что для любых формул A, B доказуемы импликации

$$\Box(A \to B) \to (\Box A \to \Box B) \tag{31}$$

и

$$\Box A \to \Box\Box A \tag{32}$$

Отметим, что

- (31) можно рассматривать как формализацию в $\mathcal{T}$ правила modus ponens, и
- (32) можно рассматривать как формализацию утверждения леммы 3.

Если доказуема импликация $A \to B$, то по лемме 3 доказуема формула $\Box(A \to B)$, откуда, на основании доказуемости формул вида (31), заключаем, что доказуема импликации $\Box A \to \Box B$.

Для обоснования доказуемости эквиваленции (30) мы докажем, что доказуемы импликации $\Box\varphi \to \Box\mathbf{0}$ и $\Box\mathbf{0} \to \Box\varphi$.

1. Поскольку формула $\mathbf{0} \to \varphi$ является аксиомой (из первой группы), т.к.
   - она имеет вид $(\mathbf{0} \to p)(\varphi/p)$, и
   - $\mathbf{0} \to p$ – тавтология логики высказываний

   то из вышесказанного следует, что доказуема импликация $\Box\mathbf{0} \to \Box\varphi$.
2. Поскольку формула $\varphi \to (\neg\varphi \to \mathbf{0})$ является аксиомой (из первой группы), т.к.
   - она имеет вид $(p \to (\neg p \to \mathbf{0}))(\varphi/p)$, и
   - $(p \to (\neg p \to \mathbf{0}))$ – тавтология логики высказываний

   то, следовательно, доказуема импликация

   $$\Box\varphi \to \Box(\neg\varphi \to \mathbf{0})$$

   откуда, используя доказуемость формул вида (31) и правило силлогизма, получаем доказуемость формулы

   $$\Box\varphi \to (\Box\neg\varphi \to \Box\mathbf{0}) \tag{33}$$

   Из (28) следует доказуемость формулы $\Box\varphi \to \neg\varphi$, откуда следует доказуемость формулы

   $$\Box\Box\varphi \to \Box\neg\varphi \tag{34}$$

   Из доказуемости формул $\Box\varphi \to \Box\Box\varphi$ и (34) по правилу силлогизма следует доказуемость формулы $\Box\varphi \to \Box\neg\varphi$ откуда, используя (33) и аксиомы из первой группы вида

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

   получаем доказуемость импликации $\Box\varphi \to \Box\mathbf{0}$. ■

Литература

[1] **Gödel, K.**: Über Formal Unentscheidbare Sätze der Principia Mathematica und Verwandter Systeme, I. *Monatshefte Math. Phys. 38, 173-198, 1931.*

[2] **Э.Мендельсон**: Введение в математическую логику. *М., Наука, 1976.*

[3] **А.Н.Колмогоров, А.Г.Драгалин**: Математическая логика. Дополнительные главы. *Издательство Московского Университета, 1984.*