Открытие неразрешимости 1:
Тезис Чёрча-Тьюринга

Станислав Олегович Сперанский

ФМКН СПбГУ

11 декабря 2020

Prelude: Формализация вычислимости

Неформальное, интуитивное понятие алгоритма использовалось на протяжении практически всей истории математики.

Традиционно конкретные алгоритмы представлялись в виде конечных наборов простых инструкций, которые можно было выполнять «по шагам». Однако до XX века никаких универсальных, общепринятых языков для записи алгоритмов не существовало.

Математики были уверены: если нужный алгоритм существует, мы его найдём. Но даже если так, как узнать, существует ли алгоритм, т.е. разрешима ли задача алгоритмически?

- Мы знаем алгоритм решения линейных диофантовых уравнений над целыми числами. Как насчёт полиномиальных?

— Можно попробовать найти алгоритм.

- Но может ли случиться так, что алгоритма не существует?

— Наверное. В таком случае нет смысла его искать.

- Но как мы узнаем, что алгоритма не существует?

— Можно попытаться показать, что никакой алгоритм не способен решить интересующую нас задачу.

- Но как перебрать все возможные алгоритмы?... И, вообще, что такое алгоритм как абстрактно-математическое понятие?...

Мораль

Если мы хотим уметь доказывать алгоритмическую неразрешимость, интуитивному, неформальному понятию алгоритма необходимо дать строгое математическое, формальное определение.

«Математизация» алгоритмов была осуществлена логиками в 1930х, где стоит особо отметить труды Гёделя, Тьюринга и Чёрча.

Впоследствии эти труды приведут к созданию архитектуры современного ПО и различных языков программирования.

В 1930х появились первые примеры алгоритмически неразрешимых задач, которые затем многократно применялись для доказательства неразрешимости задач в алгебре, топологии и так далее.

Некоторые обозначения

$\mathbb{N}$, или $\mathbb{N}^1$, обозначает $\{0, 1, 2, \dots\}$, т.е. множество всех натур. чисел. В частности, 0 считается натуральным числом. $\mathbb{N} \times \mathbb{N}$, или $\mathbb{N}^2$, обозначает множество всех упорядоченных пар натур. чисел; аналогично для $\mathbb{N}^3$, $\mathbb{N}^4$, ..., которые отождествляются с $\mathbb{N}^2 \times \mathbb{N}$, $\mathbb{N}^3 \times \mathbb{N}$, ...

Запись $x \in A$ означает, что:

- A — некоторое множество объектов;
- x принадлежит A, т.е. является элементом A.

Далее, $A \subseteq B$ означает, что A и B — множества объектов, причём A является подмножеством B, т.е. все элементы A принадлежат B.

Пусть A и B — множества объектов.

Запись $f :\subseteq A \to B$ означает, что f — *частичная* функция из A в B, которая может быть неопределена на некоторых элементах A. Через $\operatorname{dom} f$ обозначается область определения f, т.е. множество всех элементов A, на которых f определена:

$$\operatorname{dom} f \;:=\; \{x \in A \mid f(x) \text{ определено}\}.$$

В литературе по теории вычислимости обычно пишут $f(x)\downarrow$ вместо $x \in \operatorname{dom} f$ и $f(x)\uparrow$ — вместо $x \notin \operatorname{dom} f$.

Разумеется, $f : A \to B$ означает, что $f :\subseteq A \to B$ и $\operatorname{dom} f = A$, т.е. f — всюду определённая функция из A в B.

Пусть $A \subseteq \mathbb{N}^\ell$, где $\ell > 0$.

Характеристической функцией A называют $\chi_A : \mathbb{N}^\ell \to \mathbb{N}$, действующую по правилу

$$\chi_A(\vec{n}) := \begin{cases} 1 & \text{если } \vec{n} \in A \\ 0 & \text{иначе;} \end{cases}$$

под полухарактеристической функцией A понимают $\chi_A^* : \subseteq \mathbb{N}^\ell \to \mathbb{N}$, действующую по правилу

$$\chi_A^*(\vec{n}) := \begin{cases} 1 & \text{если } \vec{n} \in A \\ \uparrow & \text{иначе,} \end{cases}$$

где $\uparrow$ означает «неопределена», т.е. $\operatorname{dom} \chi_A^*$ совпадает с A.

Под р-машиной будет пониматься конечная последовательность

$$\Pi = \langle C_0, \ldots, C_k \rangle,$$

где каждая компонента имеет вид

$$\texttt{INC}\ i \quad \text{или} \quad \texttt{DEC}\ i, j.$$

Здесь INC и DEC происходят от ‘increase’ и ‘decrease’ соответственно. Компоненты Π называются её командами, а натуральные числа «i» в них — номерами её регистров.

Пусть Π — произвольная р-машина. Для удобства положим

$$m := \text{наибольший номер регистра в } \Pi.$$

Под конфигурацией понимается конечная последовательность

$$s = \langle n, r_0, \ldots, r_m \rangle,$$

где n, r_0, $\ldots$, r_m суть натуральные числа. Интуитивно:

- n — содержание счётчика команд;
- r_0, $\ldots$, r_m — содержания регистров с номерами 0, $\ldots$, m.

В случае, когда $n > k$, конфигурация s называется терминальной, или конечной; при этом счётчик команд как бы переполнен.

Для каждой нетерминальной конфигурации $s = \langle n, r_0, \ldots, r_m \rangle$, мы определим конфигурацию s^{Π} следующим образом.

- Если $C_n = \texttt{INC}\ i$, то

$$s^{\Pi} := \langle n+1, r_0, \ldots, r_{i-1}, r_i + 1, r_{i+1}, \ldots, r_m \rangle,$$

т.е. мы увеличиваем r_i на 1 и заносим $n+1$ в счётчик команд.

- Если $C_n = \texttt{DEC}\ i, j$, причём $r_i > 0$, то

$$s^{\Pi} := \langle j, r_0, \ldots, r_{i-1}, r_i - 1, r_{i+1}, \ldots, r_m \rangle,$$

т.е. мы уменьшаем r_i на 1 и заносим j в счётчик команд.

- Если $C_n = \texttt{DEC}\ i, j$, причём $r_i = 0$, то

$$s^{\Pi} := \langle n+1, r_0, \ldots, r_m \rangle,$$

т.е. мы просто заносим $n+1$ в счётчик команд.

Ясно, что последовательность

$$s, \quad s^{\Pi}, \quad s^{\Pi\Pi}, \quad s^{\Pi\Pi\Pi}, \quad \ldots \qquad (\star)$$

либо обрывается на какой-нибудь терминальной конфигурации, либо бесконечна. В первом случае мы полагаем

$$\Pi(s) \;:=\; \text{терминальная конфигурация в } (\star);$$

во втором случае считается, что $\Pi(s)$ не определено. Таким образом, Π *задаёт частичную функцию на конфигурациях*.

Замечание

Так как Π не меняет содержание регистров с номерами больше m, в определении конфигурации вместо m можно взять любое $m' > m$.

Пусть $f : \subseteq \mathbb{N}^\ell \to \mathbb{N}$, где $\ell > 0$.

Говорят, что П вычисляет f, если для всех $\vec{n} \in \mathbb{N}^\ell$:

- если $\vec{n} \in \operatorname{dom} f$, то $\Pi\left(\langle 0, 0, \vec{n}, 0^{m-\ell}\rangle\right) = \langle n, f\left(\vec{n}\right), 0^m\rangle$, где $n > k$;
- если $\vec{n} \notin \operatorname{dom} f$, то $\Pi\left(\langle 0, 0, \vec{n}, 0^{m-\ell}\rangle\right)$ не определено.

Разумеется, f называется вычислимой посредством р-машины, если существует р-машина П, которая вычисляет f.

Замечание

Здесь мы предполагаем, что $\ell + 1 \leqslant m$; иначе m нужно заменить на достаточно больше m'.

Примеры

Обозначим через GOTO j следующую p-машину:

$$\begin{array}{ll} 0: & \texttt{INC}\ 0 \\ 1: & \texttt{DEC}\ 0, j \end{array}$$

Она заносит j в счётчик команд, не меняя (исходного) содержания регистров по итогам работы.

Мы будем использовать простые p-машины в качестве «макросов» при построении более сложных p-машин; при этом макросы всегда можно эффективным образом элиминировать.

Функцию сложения на $\mathbb{N}$ можно вычислить так:

```
0: GOTO 2
1: INC 0
2: DEC 1, 1
3: DEC 2, 1
```

```
0: INC 0
1: DEC 0, 3
2: INC 0
3: DEC 1, 2
4: DEC 2, 2
```

В дальнейшем для вычисления функции умножения на $\mathbb{N}$ нам пригодится макрос ADD j TO i, где j отлично от i:

```
0: GOTO 2
1: INC i
2: DEC j, 1
```

```
0: INC 0
1: DEC 0, 3
2: INC i
3: DEC j, 2
```

Он добавляет содержание регистра j к содержанию регистра i; при этом регистр j обнуляется.

Нам также нужен макрос COPY $i, j; k$, где i, j, k попарно различны:

```
0: GOTO 3
1: INC j
2: INC k
3: DEC i, 1
4: GOTO 6
5: INC i
6: DEC k, 5
```

```
0: INC 0
1: DEC 0, 4
2: INC j
3: INC k
4: DEC i, 2
5: INC 0
6: DEC 0, 8
7: INC i
8: DEC k, 7
```

Он копирует содержимое регистра i в регистр j, используя регистр k в качестве вспомогательного.

Теперь функцию умножения на $\mathbb{N}$ можно вычислить так:

```
0:  GOTO 3
1:  COPY 1, 3; 4
2:  ADD 3 TO 0
3:  DEC 2, 1
4:  DEC 1, 4        %обнуляем регистр 1
```

Разумеется, используемые здесь макросы могут быть эффективным образом элиминированы.

Можно продолжить программировать всё более сложные функции в терминах p-машин, но мы ограничимся примерами выше.

Тезис Чёрча–Тьюринга

Вместо регистровых машин или их модификаций для формализации вычислимости можно использовать другие модели, такие как:

- «рекурсивные функции»; %Гёдель, Клини
- машины Тьюринга; %Тьюринг
- нетипизированные λ-термы; %Чёрч
- нормальные алгорифмы Маркова. %Марков

На самом деле, модели вычислимости весьма разнообразны. Они отражают различные взгляды на природу алгоритмов.

Теорема

Для всякой $f : \subseteq \mathbb{N}^\ell \to \mathbb{N}$ следующие условия эквивалентны:

- *f вычислима посредством регистровой машины;*
- *f является «рекурсивной функцией»;*
- *f вычислима посредством машины Тьюринга;*
- *f определима посредством нетипизированного λ-терма;*
- *f определима посредством нормального алгорифма Маркова.*

(Список можно продолжить, разумеется.)

Говорят, что $f : \subseteq \mathbb{N}^\ell \to \mathbb{N}$ вычислима, если можно найти алгоритм $\mathcal{A}$ такой, что для всех $\vec{n} \in \mathbb{N}^\ell$:

- если $f(\vec{n}) \downarrow$, то $\mathcal{A}$ на входе $\vec{n}$ останавливается и выдаёт $f(\vec{n})$;
- если $f(\vec{n}) \uparrow$, то $\mathcal{A}$ «зависает» на входе $\vec{n}$.

Замечание

Здесь под алгоритмами понимаются p-машины, например.

$A \subseteq \mathbb{N}^\ell$ называется разрешимым, если χ_A вычислима, и полуразрешимым, если χ_A^* вычислима. При этом нетрудно убедиться, что из разрешимости следует полуразрешимость.

В рассуждениях мы будем свободно использовать:

Тезис Чёрча–Тьюринга

$f : \subseteq \mathbb{N}^\ell \to \mathbb{N}$ интуитивно вычислима тогда и только тогда, когда она вычислима посредством регистровой машины.

Этот тезис невозможно ни доказать, ни опровергнуть математически ввиду того, что здесь участвуют одновременно:

- неформ. (немат.) понятие интуитивно вычислимой функции;
- форм. (мат.) понятие вычислимой на р-машине функции.

Статус тезиса Чёрча–Тьюринга отражает наши представления о том, как математическая модель алгоритма связана с «реальностью».

Грубо говоря, если подходить к вопросу эмпирически:

- никто не смог придумать алгоритма в традиционном, интуитивном его понимании, который невозможно смоделировать посредством подходящей p-машины.

Если же подходить к вопросу чисто математически:

- все известные формализации концепции алгоритма/понятия вычислимости равносильны: по алгоритмам одного рода мы можем эффективно строить алгоритмы другого рода, вычисляющие те же частичные функции из $\mathbb{N}^\ell$ в $\mathbb{N}$.

Наконец, тезис Чёрча–Тьюринга приобретает особую актуальность, когда речь заходит об алгоритмической *неразрешимости*.