

Математическая логика и логическое программирование

mk.cs.msu.ru → Лекционные курсы
→ Математическая логика и логическое программирование (3-й поток)

Блок 41

Задачи и проблемы
Алгоритмы
Разрешимость
М-сводимость

Лектор:
Подымов Владислав Васильевич
E-mail:
valdus@yandex.ru

ВМК МГУ, 2024/2025, осенний семестр

Вступление

Корректность и полнота метода семантических таблиц:

$\models \varphi \Leftrightarrow$ для таблицы $\langle \mid \varphi \rangle$ **существует** успешный вывод

Корректность и полнота метода резолюций:

$\models \varphi \Leftrightarrow$ для системы дизъюнктов ... **существует** успешный вывод

А можно ли написать программу, которая сможет автоматически проверить общезначимость **любой** формулы φ логики предикатов?

Оказывается, что написать такую программу **невозможно**, и пришла пора это строго сформулировать и обосновать (**теорема Чёрча**)

Аналогичные утверждения для других задач, как и большая часть сопутствующих определений, известны вам из других курсов, но всё же полезно будет всё это повторить ещё раз

Задачи и проблемы

У Васи есть 5 яблок. 2 яблока он отдал Коле.
Сколько яблок осталось у Васи?

Это пример того, что принято называть **задачей**

Более точно, это **индивидуальная задача**:

задача с конкретными входными данными и конкретным ответом (3)

У васи есть N яблок. K яблок ($K \leq N$) он отдал Коле.
Сколько яблок осталось у Васи?

Это также пример того, что принято называть **задачей**

Ответ к этой задаче определяется
значениями **параметров** (**входными данными**) N и K

Такие (*параметризованные*) задачи принято называть
массовыми задачами, или, по-другому, **проблемами**

Задачи и проблемы

Массовую задачу $\mathfrak{I}$ можно понимать как отображение $\mathfrak{I} : \mathfrak{I} \rightarrow \mathfrak{O}$, где

- ▶ $\mathfrak{I}$ — множество всевозможных **входных данных** (**входов**)
- ▶ $\mathfrak{O}$ — множество всевозможных **выходных данных** (**выходов**; **ответов**)
- ▶ значение $\mathfrak{I}(i)$ — **правильный ответ** для входа i

Например, последняя задача о яблоках — это отображение $\mathfrak{I} : \{(n, k) \mid n, k \in \mathbb{N}_0, k \leq n\} \rightarrow \mathbb{N}_0$, где $\mathfrak{I}(n, k) = n - k$

Алгоритмы и разрешимость

Алгоритм — это особая совокупность действий,¹ согласно которой **входы** заданного множества $\mathcal{I}$ преобразуются в **ответы** заданного множества $\mathcal{O}$ (или не преобразуются, если применяется бесконечно много действий)²

Алгоритмом $\mathcal{A}$ **реализуется** *частично определённое* отображение $\overline{\mathcal{A}}: \mathcal{I} \rightarrow \mathcal{O}$ следующего вида:

- ▶ если к входу i применяется конечное число действий, то $\overline{\mathcal{A}}(i)$ — ответ, вычисляемый алгоритмом
 - ▶ и алгоритм **останавливается (завершается)** на входе i
- ▶ иначе значение $\overline{\mathcal{A}}(i)$ не определено
 - ▶ и алгоритм **не останавливается (не завершается)** на входе i

1 Вспоминайте из других курсов, какая именно совокупность каких действий

2 На самом деле бывают и другие алгоритмы, согласно которым выходные данные получаются многократно, или постоянно, или понятие выходных данных отсутствует — но не будем всё переусложнять

Алгоритмы и разрешимость

Алгоритмы¹ придумываются для того, чтобы **решать** **массовые задачи**

Алгоритм $\mathcal{A}$ **решает** задачу $\mathfrak{T} : \mathfrak{I} \rightarrow \mathfrak{O}$, если $\overline{\mathcal{A}} = \mathfrak{T}$, то есть

- ▶ $\mathcal{A}$ и $\mathfrak{T}$ определены
для одинаковых множеств входных и выходных данных, и
- ▶ $\mathcal{A}$ завершается на любом входе
и всегда вычисляет правильный ответ к задаче $\mathfrak{T}$

Массовая задача (**алгоритмически**) **разрешима**,
если существует алгоритм, решающий эту задачу,
и **неразрешима**, если такого алгоритма не существует

¹ Как минимум такие алгоритмы, как на предыдущем слайде

М-сводимость

Задача распознавания — это массовая задача

с множеством ответов **{да, нет}** (оно же $\{1, 0\}$, оно же $\{\text{t}, \text{f}\}$)

Задача $\mathfrak{T}_1 : \mathfrak{I}_1 \rightarrow \{1, 0\}$ **m-сводится** (или **m-сводима**)¹

к задаче $\mathfrak{T}_2 : \mathfrak{I}_2 \rightarrow \{1, 0\}$, если существует алгоритм $\mathcal{A}$, такой что

- ▶ $\bar{\mathcal{A}} : \mathfrak{I}_1 \rightarrow \mathfrak{I}_2$ — всюду определённое отображение и
- ▶ для любого входа i задачи $\mathfrak{T}_1$ верно $\mathfrak{T}_1(i) = \mathfrak{T}_2(\mathcal{A}(i))$

Обозначенный алгоритм $\mathcal{A}$ **сводит** задачу $\mathfrak{T}_1$ к задаче $\mathfrak{T}_2$

Например, если

$\mathfrak{T}_1$ — задача проверки чётности целого числа

$$(\mathfrak{T}_1(x) = 1 \Leftrightarrow \exists k : x = 2k) \text{ и}$$

$\mathfrak{T}_2$ — задача проверки делимости целого числа на 8

$$(\mathfrak{T}_2(x) = 1 \Leftrightarrow \exists k : x = 8k), \text{ то}$$

алгоритм $\mathcal{A}$ умножения целого числа на 4 ($\bar{\mathcal{A}}(x) = 4x$) m-сводит $\mathfrak{T}_1$ к $\mathfrak{T}_2$:

$$\mathfrak{T}_1(x) = \mathfrak{T}_2(4x) = \mathfrak{T}_2(\bar{\mathcal{A}}(x))$$

¹ Есть много разных видов сводимости, и этот вид (*должен быть*) вам известен из рассказа про останов и самоприменимость в курсе, посвящённом алгоритмам.

Других видов здесь не будет, так что о смысле «m» можно не задумываться

М-сводимость

Теорема (об m -сводимости). Если задача $\mathfrak{T}_1$ m -сводится к разрешимой задаче $\mathfrak{T}_2$, то задача $\mathfrak{T}_1$ также разрешима

Доказательство.¹

Положим, что задача $\mathfrak{T}_2$ разрешима и что задача $\mathfrak{T}_1$ m -сводится к $\mathfrak{T}_2$

По **определению разрешимости**, существует алгоритм $\mathcal{A}$, решающий $\mathfrak{T}_2$

По **определению m -сводимости**,

существует алгоритм $\mathcal{B}$, сводящий $\mathfrak{T}_1$ к $\mathfrak{T}_2$

Тогда существует и алгоритм решения задачи $\mathfrak{T}_1$:

последовательно применим $\mathcal{B}$ и $\mathcal{A}$ к входу i задачи $\mathfrak{T}_1$ $(i \xrightarrow{\mathcal{B}} j \xrightarrow{\mathcal{A}} o)$

По выбору алгоритма $\mathcal{A}$, $\overline{\mathcal{A}}(\overline{\mathcal{B}}(i)) = \mathfrak{T}_2(\overline{\mathcal{B}}(i))$

По выбору алгоритма $\mathcal{B}$, $\mathfrak{T}_2(\overline{\mathcal{B}}(i)) = \mathfrak{T}_1(i)$ ▼

Следствие. Если неразрешимая задача $\mathfrak{T}_1$ m -сводится к задаче $\mathfrak{T}_2$, то задача $\mathfrak{T}_2$ также неразрешима

¹ Это доказательство (должно быть) вам известно из курса, посвящённого алгоритмам. Повторим его «для профилактики».

М-сводимость

Иллюстрация теоремы

Мой сосед умеет проверять (алгоритмом $\mathcal{A}$),
делится ли целое число нацело на 8:

$$\mathfrak{T}_2(x) = 1 \Leftrightarrow x = 8k$$

Я хочу научиться решать задачу проверки чётности целого числа:

$$\mathfrak{T}_1(x) = 1 \Leftrightarrow x = 2m$$

Тогда я могу организовать алгоритм решения своей задачи $\mathfrak{T}_1$ так:

1. Умножить входное число на 4 подходящим алгоритмом $\mathcal{B}$:

$$\overline{\mathcal{B}}(x) = 4x$$

2. Спросить у соседа, делится ли полученное число на 8:

$$\mathfrak{T}_2(\overline{\mathcal{B}}(x)) = \overline{\mathcal{A}}(\overline{\mathcal{B}}(x)) = ?$$

3. Получив ответ от соседа, выдать его за свой:

$$\overline{\mathcal{A}}(\overline{\mathcal{B}}(x)) = \mathfrak{T}_1(x)$$

М-сводимость

Иллюстрация следствия

Мой сосед знает, что
проблема самоприменимости машин Тьюринга ($\mathfrak{T}_1$) неразрешима

Я хочу научиться решать проблему останова машин Тьюринга ($\mathfrak{T}_2$)

Сосед

- ▶ убеждает меня в неразрешимости $\mathfrak{T}_1$ и
- ▶ объясняет, как по произвольному входу i к своей задаче построить вход $\bar{B}(i)$ к моей задаче, такой что

$$\mathfrak{T}_1(i) = \mathfrak{T}_2(\bar{B}(i))$$

Если бы вдруг я научился решать $\mathfrak{T}_2$,
то мой сосед научился бы решать $\mathfrak{T}_1$ с моей помощью так,
как это обозначено в иллюстрации теоремы

Значит, моя задача $\mathfrak{T}_2$ неразрешима