

Математическая логика и логическое программирование

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

Блок 43

Логические программы:
встроенные предикаты и функции

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

ВМК МГУ, 2024/2025, осенний семестр

Вступление

При помощи ХЛП можно решать любые разрешимые задачи, но всё же чего-то не хватает

Например, если хочется написать ХЛП, складывающую два числа из \mathbb{N}_0 , то это не получится сделать сильно проще, чем:

1. Использовать представление чисел списками нолей и единиц, хранящими двоичную запись с младшим разрядом в голове:
 - ▶ Например, $13 = (1011)_2 = \mathbf{1.1.0.1.nil}$
2. Для операций использовать соответствующие предикаты
 - ▶ Например, предикат сложения: $\text{plus}(X, Y, Z) = \text{t} \Leftrightarrow Z = X + Y$
3. Реализовать необходимые предикаты для выбранного представления:

```
plus(nil, L, L);  
plus(L, nil, L);  
plus(0.L, X.M, X.N)  $\leftarrow$  ...;  
plus(1.L, X.M, Y.N)  $\leftarrow$  ...;  
...
```

Выходит не очень удобно для использования — а нельзя ли лучше?

Вступление

Дальше (в этом блоке слайдов и в следующих) будут обсуждаться конструкции, которые содержатся в языке Prolog и существенно повышают удобство практического использования логических программ

Но введение этих конструкций сопряжено с техническими и идеологическими трудностями, из-за которых будем иногда

- ▶ «забывать» про декларативную семантику и обсуждать только операционную
- ▶ «вспоминать» про декларативную семантику и пытаться преодолеть трудности, иногда полноценно, а иногда только отчасти

Чтобы подчеркнуть расхождение между тем, что будет рассказываться про логические программы дальше, и тем, что рассказывалось до сих пор, не будем называть программы с дальнейшими добавлениями и уточнениями «хорновскими», называя их просто «**логические программы**»

Встроенные предикаты и функции

В языке логических программ на практике содержатся **встроенные предикаты и функции**: предикатные и функциональные символы, имеющие предзаданный смысл, выходящий за рамки SLD-резолюции, и предназначенные для записи в телах правил и запросов

Смысл встроенных предикатов будет обсуждаться **только** в рамках операционной семантики, и он будет задаваться как сочетание **критерия выполнимости** и **унификатора**:

- ▶ Если в качестве подцели выбран не встроенный предикат, то шаг вычисления состоит в обычном построении SLD-резольвенты
- ▶ Иначе:
 - ▶ Если выполнен **критерий выполнимости** встроенного предиката, выбранного в качестве подцели, то
 - ▶ этот предикат удаляется из запроса, и
 - ▶ к оставшейся части запроса применяется **унификатор**
 - ▶ Иначе построенное вычисление считается **тупиковым** (невозможно выполнить следующий шаг)

Типы данных

В «удобных» языках программирования, как правило, есть хотя бы минимальный набор **типов данных**

Для примера подробно обсудим тип **integer** целых чисел:
 $0, 1, (-1), 2, (-2), \dots$

С точки зрения логики предикатов,

- ▶ $0, 1, (-1), 2, (-2), \dots$ — это целочисленные **константы**
- ▶ **integer** — это одноместный предикатный символ: $\text{integer}(x) = \langle x \text{ — целое число} \rangle$

Типы данных: integer

`integer` в логических программах — это встроенный одноместный предикат:

- ▶ Критерий выполнимости $\text{integer}(t)$: t — это целочисленная константа
- ▶ Унификатор: ϵ

Например:

$?integer(3), p(X)$

$\downarrow \epsilon$

$?p(X)$

$?integer(0.nil)$

тупик

$?integer(1 + 2)$

тупик

$?integer(X)$

тупик

Типы данных: integer , $+$, $-$, \cdot ($*$), $/$ (div), $\%$ (mod)

Для целых чисел можно естественно ввести

- ▶ операции: $+$, $-$, \cdot (она же $*$), $/$ (она же **div**), $\%$ (она же **mod**), ...
 - ▶ С точки зрения логики предикатов, это функциональные символы подходящей местности в инфиксной записи с естественным смыслом
- ▶ отношения: $<$, \leq , $>$, \geq , ...
 - ▶ С точки зрения логики предикатов, это предикатные символы подходящей местности в инфиксной записи с естественным смыслом
 - ▶ Особое место в этом списке занимает отношение равенства, его обсудим отдельно позже

Небольшая поправка: знаки операций и отношений, используемые в Prolog, могут отличаться, читайте документацию

Типы данных: integer, <, ≤, >, ≥

Отношение $\bowtie \in \{<, \leq, >, \geq\}$ над целыми числами в логических программах — это встроенный двуместный предикат в инфиксной записи:

- ▶ Критерий выполнимости $t_1 \bowtie t_2$: t_1 и t_2 — целочисленные константы, входящие в отношение \bowtie
- ▶ Унификатор: ϵ

Например:

?1 < 3, p(X)

↓ ϵ

?p(X)

?3 < 1

тупик

?X < 2

тупик

?1 < 1 + 2

тупик

Как видно по самому правому и самому левому примерам, выражение **1 + 2** расценивается программой не как число **3**, а именно как выражение — запись, сама по себе не являющаяся числом

Чтобы **вычислить** это выражение (преобразовать его в число), требуется применить соответствующий предикат

Вычисляющий предикат (is)

is — это встроенный двуместный предикат (записывающийся инфиксно), предназначенный для вычисления значения выражения с записью в переменную:

- ▶ Критерий выполнимости $t_1 \text{ is } t_2$:
 - ▶ t_1 — переменная и
 - ▶ t_2 — выражение (терм), имеющее значение (без переменных и построенное корректно относительно типов)
- ▶ Унификатор: $\{t_1/val\}$, где val — значение выражения t_2

Например:

$?X \text{ is } \mathbf{1 + 2}, p(X), r(Y)$	$? \mathbf{3} \text{ is } \mathbf{1 + 2}$	$?X \text{ is } \mathbf{1 + Y}$	$? \mathbf{1 + 2} \text{ is } \mathbf{3}$	$? \mathbf{1 + 2} \text{ is } X$
$\downarrow \{X/\mathbf{3}\}$	тупик	тупик	тупик	тупик
$?p(\mathbf{3}), r(Y)$				

Небольшая поправка:

- ▶ В интерпретаторе языка Prolog с немалой вероятностью во втором слева примере будет не тупик, а шаг вычисления с унификатором ε
- ▶ Выше изложена семантика **is** в «исходном» варианте языка Prolog, а остальное здесь считается «отклонением от стандарта»

Предикаты равенства и неравенства ($=$, $==$, $\backslash=$, $=\backslash=$)

В Prolog используется два вида равенства термов:

- ▶ **Сильное**: посимвольное совпадение
- ▶ **Слабое**: унифицируемость

$==$ — встроенный двуместный предикат сильного равенства в инфиксной записи:

- ▶ Критерий выполнимости $t_1 == t_2$: термы t_1 и t_2 посимвольно совпадают
- ▶ Унификатор: ϵ

Например:

$$?X + 1 == X + 1, p(X)$$

$\downarrow \epsilon$

$$?p(X)$$

$$?X == 1 + 2$$

тупик

$$?3 == 1 + 2$$

тупик

$$?X + 1 == 1 + X$$

тупик

$$?X + 1 == 2 + X$$

тупик

$$?1 + 2 == 2 + 1$$

тупик

Предикаты равенства и неравенства ($=$, $==$, \neq , \neq)

\neq — встроенный двуместный предикат сильного неравенства в инфиксной записи:

- ▶ Критерий выполнимости $t_1 \neq t_2$: t_1 и t_2 не совпадают посимвольно
- ▶ Унификатор: ϵ

Например:

$$?X + 1 \neq X + 1$$

тупик

$$?X \neq 1 + 2, p(X)$$
$$\downarrow \epsilon$$
$$?p(X)$$

$$?3 \neq 1 + 2, p(X)$$
$$\downarrow \epsilon$$
$$?p(X)$$

$$?X + 1 \neq 1 + X, p(X)$$
$$\downarrow \epsilon$$
$$?p(X)$$

$$?X + 1 \neq 2 + X, p(X)$$
$$\downarrow \epsilon$$
$$?p(X)$$

$$?1 + 2 \neq 2 + 1, p(X)$$
$$\downarrow \epsilon$$
$$?p(X)$$

Предикаты равенства и неравенства ($=$, $==$, \neq , \neq)

$=$ — встроенный двуместный предикат слабого равенства в инфиксной записи:

- ▶ Критерий выполнимости $t_1 = t_2$: $\mathcal{U}(t_1, t_2) \neq \emptyset$
- ▶ Унификатор: какой-либо наиболее общий унификатор t_1 и t_2

Например:

$$?X + 1 = X + 1, p(X)$$

$$\downarrow \varepsilon$$
$$?p(X)$$

$$?X = 1 + 2, p(X), r(Y)$$

$$\downarrow \{X/1 + 2\}$$
$$?p(1 + 2), r(Y)$$

$$?3 = 1 + 2$$

тупик

$$?X + 1 = 1 + X, p(X), r(Y)$$

$$\downarrow \{X/1\}$$
$$?p(1), r(Y)$$

$$?X + 1 = 2 + X$$

тупик

$$?1 + 2 = 2 + 1$$

тупик

Предикаты равенства и неравенства ($=$, $==$, $\backslash=$, $=\backslash=$)

$\backslash=$ — встроенный двуместный предикат слабого неравенства в инфиксной записи:

- ▶ Критерий выполнимости $t_1 \backslash= t_2$: $\mathcal{V}(t_1, t_2) = \emptyset$
- ▶ Унификатор: ϵ

Например:

$$\begin{array}{c} ?X + 1 \backslash= X + 1 \\ \text{тупик} \end{array}$$

$$\begin{array}{c} ?X \backslash= 1 + 2 \\ \text{тупик} \end{array}$$

$$\begin{array}{c} ?3 \backslash= 1 + 2, p(X) \\ \downarrow \epsilon \\ ?p(X) \end{array}$$

$$\begin{array}{c} ?X + 1 \backslash= 1 + X \\ \text{тупик} \end{array}$$

$$\begin{array}{c} ?X + 1 \backslash= 2 + X, p(X) \\ \downarrow \epsilon \\ ?p(X) \end{array}$$

$$\begin{array}{c} ?1 + 2 \backslash= 2 + 1, p(X) \\ \downarrow \epsilon \\ ?p(X) \end{array}$$