Математическая логика и логическое программирование

Лектор:

Подымов Владислав Васильевич

e-mail:
valdus@yandex.ru

2015, весенний семестр

Лекция 14

Правила выбора подцели

Деревья SLD-резолютивных вычислений

Правила поиска

Стратегии вычисления

Что имеется на данный момент?

- Хорновские логические программы:
  - программа — это совокупность **правил** и **фактов** $(A_0 \leftarrow A_1, \ldots, A_k;)$
  - к программе посылается **запрос** $(?C_1, \ldots, C_m)$
- Декларативная семантика программ:
  - **правильный ответ** — это набор значений целевых переменных, для которого запрос логически следует из программы
- Операционная семантика программ:
  - **вычисленный ответ** — это набор значений целевых переменных, получаемый в результате построения **какого-либо** успешного SLD-резолютивного вычисления

Стратегии вычисления

В операционной семантике есть немалая свобода выбора

Например,

(1) Летает(**самолёт**);
(2) Летает(**орёл**);
(3) Птица(**орёл**);
(4) Птица(**пингвин**);

1. Какую подцель обработать?
2. Какое утверждение использовать?

Правило выбора подцели

Ответ на первый вопрос начнём с примера:

(1) Летает(**самолёт**);
(2) Летает(**орёл**);
(3) Птица(**орёл**);
(4) Птица(**пингвин**);

Есть два способа получить вычисленный ответ:

Правило выбора подцели

Ответ на первый вопрос начнём с примера:

(1) Летает(**самолёт**);
(2) Летает(**орёл**);
(3) Птица(**орёл**);
(4) Птица(**пингвин**);

Есть два способа получить вычисленный ответ:

?Летает(X); Птица(X)
↓(2); {X/**орёл**}
?Птица(**орёл**)
↓(3); ε
□

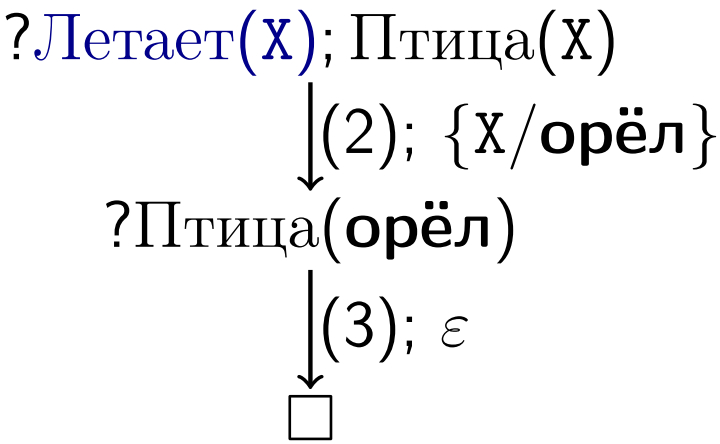
- Получили вычисленный ответ {X/**орёл**}

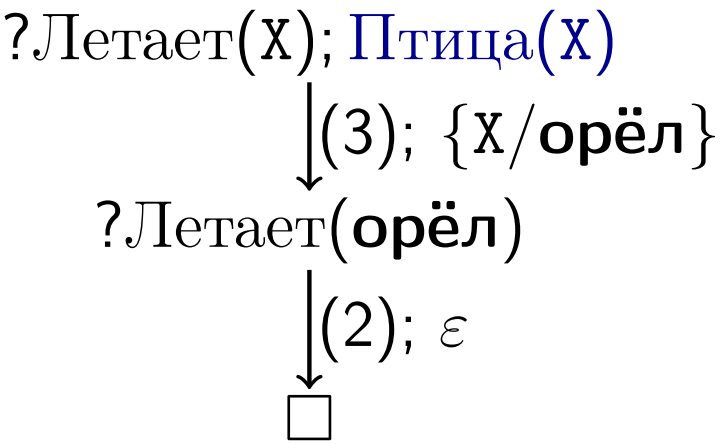
Правило выбора подцели

Ответ на первый вопрос начнём с примера:

(1) Летает(**самолёт**);
(2) Летает(**орёл**);
(3) Птица(**орёл**);
(4) Птица(**пингвин**);

Есть два способа получить вычисленный ответ:





- Получили вычисленный ответ {X/**орёл**}
- Получили вычисленный ответ {X/**орёл**}

Правило выбора подцели

В последнем примере (единственный) вычисленный ответ $\{\mathtt{X}/\textbf{орёл}\}$ можно было получить при любом порядке выбора подцелей

Возникает естественное предположение:

вычислимость ответа не зависит от порядка выбора подцелей

Аргумент за: **для получения ответа нужно решить каждую подзадачу, записанную в запросе, — и какая разница, в каком порядке их решать?**

Аргумент против: **ответ, найденный при решении одной подзадачи, может помочь решить другую**

Оказывается, аргумент “против” несостоятелен, и само предположение верно

Правило выбора подцели

Правило выбора подцели $\mathfrak{R}$ программе $\mathcal{P}$ и частичному SLD-резолютивному вычислению $\overline{\mathfrak{C}}$, оканчивающемуся запросом $?C_1, \ldots, C_m$, сопоставляет индекс i обрабатываемой подцели C_i

Вычисление, в котором обрабатываемая подцель всегда выбирается по правилу $\mathfrak{R}$, называется $\mathfrak{R}$-вычислением

Ответ, полученный в результате успешного $\mathfrak{R}$-вычисления, называется $\mathfrak{R}$-вычисленным ответом

Покажем, что выбор правила $\mathfrak{R}$ не влияет то, какие ответы можно вычислить

Правило выбора подцели

Теорема сильной полноты операционной семантики

Для любого правила выбора подцели $\mathfrak{R}$ и любого правильного ответа θ на запрос $\mathfrak{C}$ к программе $\mathcal{P}$ существует $\mathfrak{R}$-вычисленный ответ η на запрос $\mathfrak{C}$ к программе $\mathcal{P}$, такой что равенство

$$\theta = \eta\rho$$

справедливо для некоторой подстановки ρ

Доказательство.

Полнота операционной семантики: существует вычисленный ответ η' на запрос $\mathfrak{C}$ к программе $\mathcal{P}$ и подстановка ρ', такие что $\theta = \eta'\rho'$

Этот ответ получается в результате некоторого успешного SLD-резолютивного вычисления $(Cl_1, \eta'_1, \mathfrak{C}_1), \ldots, (Cl_k, \eta'_k, \square)$:

$$\eta' = \eta'_1 \ldots \eta'_k$$

Перестроим это вычисление так, чтобы оно стало $\mathfrak{R}$-вычислением

Правило выбора подцели

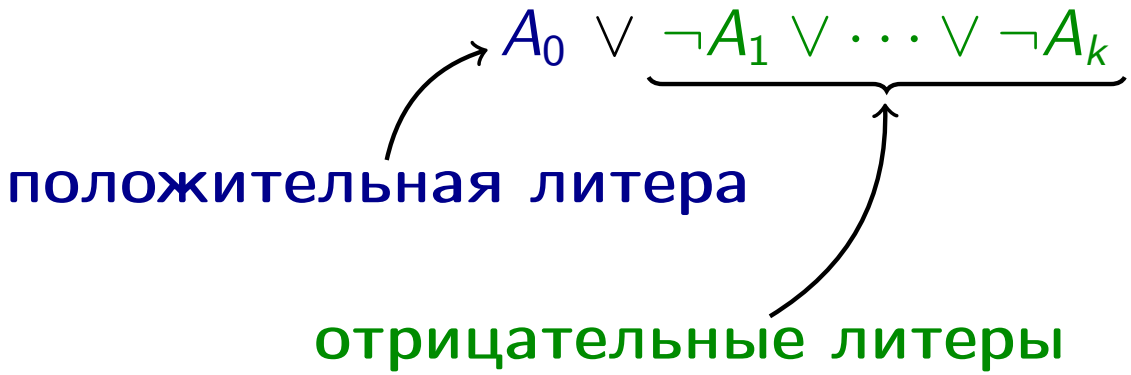
Переключательная лемма

Сначала немного обозначений

Для правила $Cl : A_0 \leftarrow A_1, \ldots, A_k$;

- **записью** Cl^+ **обозначим его заголовок** A_0,
- **а записью** Cl^- **— тело** $A_1, \ldots, A_k$

Пояснение этих обозначений:



Правило выбора подцели

Переключательная лемма

Пусть запрос $\mathfrak{C}$ к программе $\mathcal{P}$ имеет вычисление

$$\begin{array}{l} \mathfrak{C}\colon\ ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m \\ \quad\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+) \\ \mathfrak{C}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1 \\ \quad\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+) \\ \mathfrak{C}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2 \end{array}$$

Тогда он также имеет вычисление

$$\begin{array}{l} \mathfrak{C}\colon\ ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m \\ \quad\Big\downarrow Cl_2;\ \eta_1 \in \text{НОУ}(C_j, Cl_2^+) \\ \mathfrak{C}''_1\colon (?C_1, \ldots, C_i, \ldots, Cl_2^-, \ldots, C_m)\eta_1 \\ \quad\Big\downarrow Cl_1;\ \eta_2 \in \text{НОУ}(C_i\eta_1, Cl_1^+) \\ \mathfrak{C}''_2\colon (?C_1\eta_1, \ldots, Cl_1^-, \ldots, Cl_2^-\eta_1, \ldots, C_m\eta_1)\eta_2 \end{array}$$

причём запросы $\mathfrak{C}'_2$ и $\mathfrak{C}''_2$ являются вариантами друг друга

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\begin{array}{l} \mathfrak{C}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m \\ \quad \Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+) \\ \mathfrak{C}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1 \\ \quad \Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+) \\ \mathfrak{C}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2 \end{array}$$

Варианты утверждений выбираются так, чтобы их переменные были уникальны

Значит, $Cl_2^+\theta_2 = Cl_2^+\theta_1\theta_2$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{G}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{G}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{G}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

Варианты утверждений выбираются так, чтобы их переменные были уникальны

Значит, $Cl_2^+\theta_2 = Cl_2^+\theta_1\theta_2$

$$C_j\theta_1\theta_2 = Cl_2^+ \quad \theta_2$$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{C}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{C}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{C}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

Варианты утверждений выбираются так, чтобы их переменные были уникальны

Значит, $Cl_2^+\theta_2 = Cl_2^+\theta_1\theta_2$

Тогда $C_j\theta_1\theta_2 = Cl_2^+\theta_1\theta_2$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{G}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{G}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{G}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

Варианты утверждений выбираются так, чтобы их переменные были уникальны

Значит, $Cl_2^+\theta_2 = Cl_2^+\theta_1\theta_2$

Тогда $C_j\theta_1\theta_2 = Cl_2^+\theta_1\theta_2$, то есть атомы C_j и Cl_2^+ унифицируемы

Значит, существует их наиболее общий унификатор:

$$\eta_1 \in \text{НОУ}(C_j, Cl_2^+)$$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\begin{array}{l} \mathfrak{G}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m \\ \quad \Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+) \\ \mathfrak{G}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1 \\ \quad \Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+) \\ \mathfrak{G}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2 \end{array}$$

Варианты утверждений выбираются так, чтобы их переменные были уникальны

Значит, $Cl_2^+\theta_2 = Cl_2^+\theta_1\theta_2$

Тогда $C_j\theta_1\theta_2 = Cl_2^+\theta_1\theta_2$, то есть атомы C_j и Cl_2^+ унифицируемы

Значит, существует их наиболее общий унификатор:

$$\eta_1 \in \text{НОУ}(C_j, Cl_2^+), \text{ причём } \theta_1\theta_2 = \eta_1\rho$$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{G}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{G}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{G}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

$$(\eta_1 \in \text{НОУ}(C_j, Cl_2^+),\ \theta_1\theta_2 = \eta_1\rho)$$

Имеем $\quad C_i\theta_1 = Cl_1^+\theta_1$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{C}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{C}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{C}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

$$(\eta_1 \in \text{НОУ}(C_j, Cl_2^+),\ \theta_1\theta_2 = \eta_1\rho)$$

Имеем $\quad C_i\theta_1\theta_2 = Cl_1^+\theta_1\theta_2$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{C}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{C}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{C}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

$$(\eta_1 \in \text{НОУ}(C_j, Cl_2^+),\ \theta_1\theta_2 = \eta_1\rho)$$

Имеем $C_i\eta_1\rho = C_i\theta_1\theta_2 = Cl_1^+\theta_1\theta_2 = Cl_1^+\eta_1\rho$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{G}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{G}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{G}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

$$(\eta_1 \in \text{НОУ}(C_j, Cl_2^+),\ \theta_1\theta_2 = \eta_1\rho)$$

Имеем $C_i\eta_1\rho = C_i\theta_1\theta_2 = Cl_1^+\theta_1\theta_2 = Cl_1^+\eta_1\rho$

При этом $Cl_1^+\eta_1 = Cl_1^+$ (опять уникальность переменных)

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{G}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{G}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{G}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

$$(\eta_1 \in \text{НОУ}(C_j, Cl_2^+),\ \theta_1\theta_2 = \eta_1\rho)$$

Имеем $C_i\eta_1\rho = C_i\theta_1\theta_2 = Cl_1^+\theta_1\theta_2 = Cl_1^+\eta_1\rho$

При этом $Cl_1^+\eta_1 = Cl_1^+$

$$C_i\eta_1\rho = Cl_1^+\eta_1\rho$$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{G}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{G}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{G}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

$$(\eta_1 \in \text{НОУ}(C_j, Cl_2^+),\ \theta_1\theta_2 = \eta_1\rho)$$

Имеем $C_i\eta_1\rho = C_i\theta_1\theta_2 = Cl_1^+\theta_1\theta_2 = Cl_1^+\eta_1\rho$

При этом $Cl_1^+\eta_1 = Cl_1^+$

Значит, $C_i\eta_1\rho = Cl_1^+ \quad \rho$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{G}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{G}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{G}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

$$(\eta_1 \in \text{НОУ}(C_j, Cl_2^+),\ \theta_1\theta_2 = \eta_1\rho)$$

Имеем $C_i\eta_1\rho = C_i\theta_1\theta_2 = Cl_1^+\theta_1\theta_2 = Cl_1^+\eta_1\rho$

При этом $Cl_1^+\eta_1 = Cl_1^+$

Значит, $C_i\eta_1\rho = Cl_1^+ \quad \rho$, то есть атомы $C_i\eta_1$ и Cl_1^+ унифицируемы

Следовательно, существует их наиболее общий унификатор:

$$\eta_2 \in \text{НОУ}(C_i\eta_1, Cl_1^+)$$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{C}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+)$$
$$\mathfrak{C}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1$$
$$\Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+)$$
$$\mathfrak{C}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2$$

$$(\eta_1 \in \text{НОУ}(C_j, Cl_2^+),\ \theta_1\theta_2 = \eta_1\rho)$$

Имеем $C_i\eta_1\rho = C_i\theta_1\theta_2 = Cl_1^+\theta_1\theta_2 = Cl_1^+\eta_1\rho$

При этом $Cl_1^+\eta_1 = Cl_1^+$

Значит, $C_i\eta_1\rho = Cl_1^+ \quad \rho$, то есть атомы $C_i\eta_1$ и Cl_1^+ унифицируемы

Следовательно, существует их наиболее общий унификатор:

$$\eta_2 \in \text{НОУ}(C_i\eta_1, Cl_1^+),\ \text{причём}\ \rho = \eta_2\rho'$$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\begin{array}{l} \mathfrak{G}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m \\ \quad \Big\downarrow Cl_1;\ \theta_1 \in \text{НОУ}(C_i, Cl_1^+) \\ \mathfrak{G}'_1\colon (?C_1, \ldots, Cl_1^-, \ldots, C_j, \ldots, C_m)\theta_1 \\ \quad \Big\downarrow Cl_2;\ \theta_2 \in \text{НОУ}(C_j\theta_1, Cl_2^+) \\ \mathfrak{G}'_2\colon (?C_1\theta_1, \ldots, Cl_1^-\theta_1, \ldots, Cl_2^-, \ldots, C_m\theta_1)\theta_2 \end{array}$$

Что получили в результате:

- $\eta_1 \in \text{НОУ}(C_j, Cl_2^+)$
- $\eta_2 \in \text{НОУ}(C_i\eta_1, Cl_1^+)$
- $\theta_1\theta_2 = \eta_1\rho = \eta_1\eta_2\rho'$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{C}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\Big\downarrow Cl_2;\ \eta_1 \in \text{НОУ}(C_j, Cl_2^+)$$
$$\mathfrak{C}''_1\colon (?C_1, \ldots, C_i, \ldots, Cl_2^-, \ldots, C_m)\eta_1$$
$$\Big\downarrow Cl_1;\ \eta_2 \in \text{НОУ}(C_i\eta_1, Cl_1^+)$$
$$\mathfrak{C}''_2\colon (?C_1\eta_1, \ldots, Cl_1^-, \ldots, Cl_2^-\eta_1, \ldots, C_m\eta_1)\eta_2$$

Что получили в результате:

- $\eta_1 \in \text{НОУ}(C_j, Cl_2^+)$
- $\eta_2 \in \text{НОУ}(C_i\eta_1, Cl_1^+)$
- $\theta_1\theta_2 = \eta_1\rho = \eta_1\eta_2\rho'$

Тогда можно перестроить вычисление

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\mathfrak{C}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m$$
$$\downarrow Cl_2;\ \eta_1 \in \text{НОУ}(C_j, Cl_2^+)$$
$$\mathfrak{C}''_1\colon (?C_1, \ldots, C_i, \ldots, Cl_2^-, \ldots, C_m)\eta_1$$
$$\downarrow Cl_1;\ \eta_2 \in \text{НОУ}(C_i\eta_1, Cl_1^+)$$
$$\mathfrak{C}''_2\colon (?C_1\eta_1, \ldots, Cl_1^-, \ldots, Cl_2^-\eta_1, \ldots, C_m\eta_1)\eta_2$$

Что получили в результате:

- $\theta_1\theta_2 = \eta_1\rho = \eta_1\eta_2\rho'$

Правило выбора подцели

Доказательство леммы.

Имеем:

$$\begin{array}{l} \mathfrak{C}\colon ?C_1, \ldots, C_i, \ldots, C_j, \ldots, C_m \\ \quad \downarrow Cl_2;\ \eta_1 \in \text{НОУ}(C_j, Cl_2^+) \\ \mathfrak{C}''_1\colon (?C_1, \ldots, C_i, \ldots, Cl_2^-, \ldots, C_m)\eta_1 \\ \quad \downarrow Cl_1;\ \eta_2 \in \text{НОУ}(C_i\eta_1, Cl_1^+) \\ \mathfrak{C}''_2\colon (?C_1\eta_1, \ldots, Cl_1^-, \ldots, Cl_2^-\eta_1, \ldots, C_m\eta_1)\eta_2 \end{array}$$

Что получили в результате:

- $\theta_1\theta_2 = \eta_1\rho = \eta_1\eta_2\rho'$
- $\eta_1\eta_2 = \theta_1\mu = \theta_1\theta_2\mu'$

Полученное вычисление можно перестроить по той же схеме в исходное

Из равенств $\theta_1\theta_2 = \eta_1\eta_2\rho'$ и $\eta_1\eta_2 = \theta_1\theta_2\mu'$ следует, что $\mathfrak{C}'_2$ и $\mathfrak{C}''_2$ — варианты друг друга

конец доказательства леммы

Правило выбора подцели

Доказательство теоремы (завершение).

Имеем успешное SLD-резолютивное вычисление

$$(Cl_1, \eta'_1, \mathfrak{C}_1), \ldots, (Cl_k, \eta'_k, \square)$$

и вычисленный ответ $\eta' = \eta'_1 \ldots \eta'_k$

Требуется показать, что η' — $\mathfrak{R}$-вычисленный ответ (с точностью до переименования)

Будем просматривать вывод от начала к концу
Если на текущем (p-м) шаге выбор подцели C_i не согласуется с $\mathfrak{R}$:

- ищем шаг (q-й), на котором обрабатывается подцель, которую нужно обработать сейчас
  (такой шаг обязательно есть)
- применяем $(q - p)$ раз переключательную лемму, продвигая нужную подцель к началу
- получаем обработку нужной подцели на p-м шаге

Правило выбора подцели

Доказательство теоремы (завершение).

Имеем успешное SLD-резолютивное вычисление

$$(Cl_1, \eta'_1, \mathfrak{C}_1), \ldots, (Cl_k, \eta'_k, \square)$$

и вычисленный ответ $\eta' = \eta'_1 \ldots \eta'_k$

Требуется показать, что η' — $\mathfrak{R}$-вычисленный ответ (с точностью до переименования)

Просмотрев все шаги и применив много раз переключательную лемму, получим успешное $\mathfrak{R}$-вычисление и $\mathfrak{R}$-вычисленный ответ, являющийся переименованием η'

конец доказательства теоремы

(переключательная лемма говорит только о двух соседних шагах вывода; применение её к выводу целиком — **самостоятельно**)

Правило выбора подцели

Используя теорему сильной полноты операционной семантики, мы можем **без ограничения общности** выбрать конкретное правило выбора подцели

Далее всегда будем придерживаться такого правила:

в запросе всегда обрабатывается
самая левая (первая) подцель

Это стандартное правило выбора подцели

Теперь можно перейти ко второму вопросу:

как выбор программных утверждений
влияет на возможности вычисления ответов?

Выбор программных утверждений

как выбор программных утверждений
влияет на возможности вычисления ответов?

Обратимся к операционной семантике:

- подцель — это задача, которую необходимо решить
- программное утверждение — это метод решения этой задачи
- выбор программного утверждения — это выбор конкретного метода решения

Некоторые методы могут быть результативными, некоторые — нет

Выбор программных утверждений

Например,

(1) Летает(**самолёт**);
(2) Летает(**орёл**);
(3) Птица(**орёл**);
(4) Птица(**пингвин**);

?Летает(X), Птица(X)
↓(2); {X/**орёл**}
?Птица(**орёл**)
↓(3); ε
□

Успех! ☺

$\theta = \{X/\textbf{орёл}\}$

?Летает(X), Птица(X)
↓(1); {X/**самолёт**}
?Птица(**самолёт**)

Тупик ☹

Выбор программных утверждений

Как видно из примера и содержательных пояснений,

в общем случае
для получения успешного вычисления
необходимо перебрать все программные утверждения

Тогда возникает другой вопрос:

как организовать перебор программных утверждений?

Этот вопрос приводит к тому, что необходимо “хорошее” описание всевозможных SLD-резолютивных вычислений

Деревья вычислений

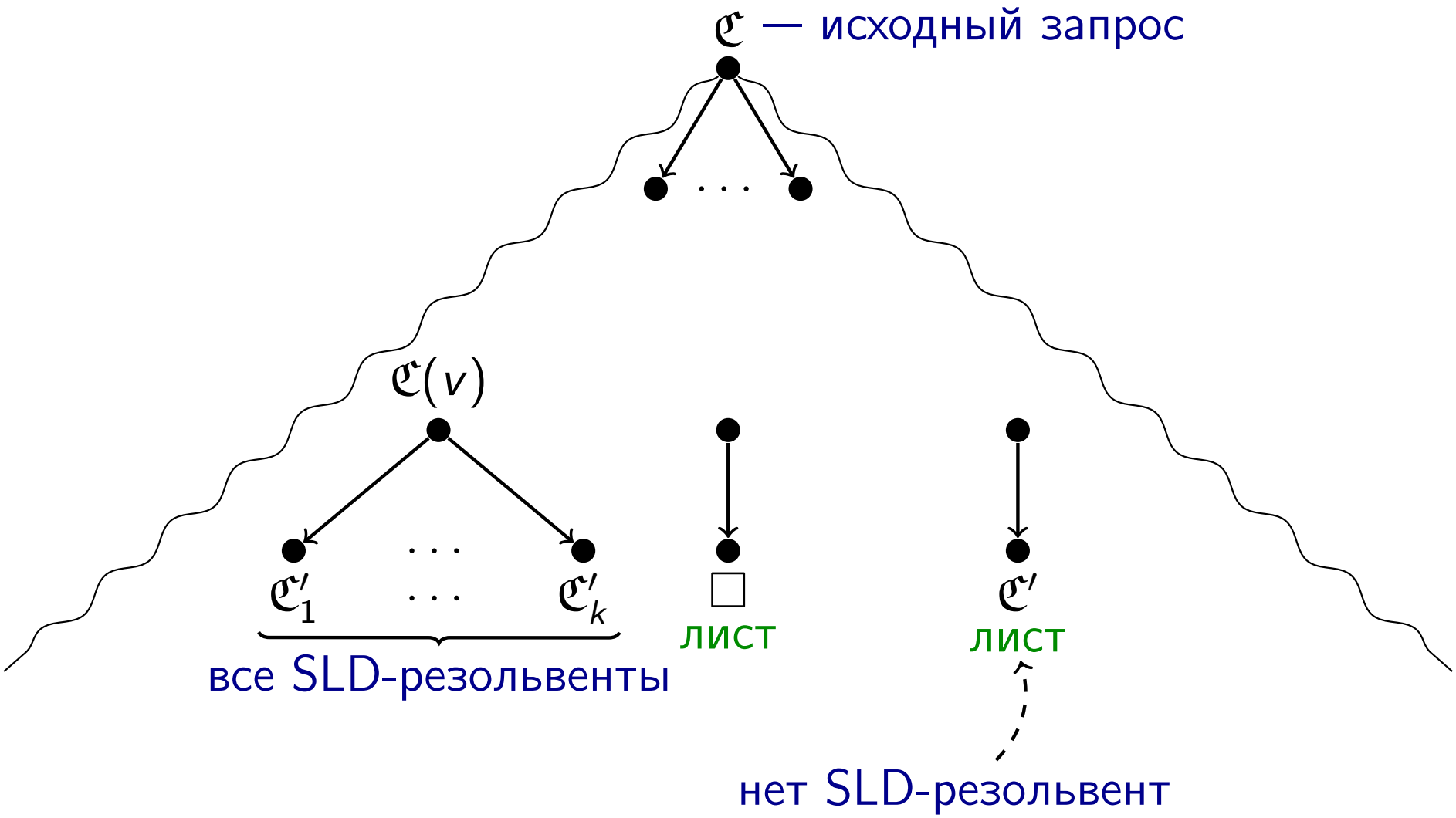
Дерево SLD-резолютивных вычислений запроса $\mathfrak{C}$ к программе $\mathcal{P}$ — это размеченное корневое дерево, удовлетворяющее следующим требованиям:

- каждая вершина v помечена некоторым запросом $\mathfrak{C}(v)$
- корень дерева помечен запросом $\mathfrak{C}$
- каждая вершина v имеет столько потомков, сколько существует SLD-резольвент запроса $\mathfrak{C}(v)$ с утверждениями $\mathcal{P}$
- потомки каждой вершины v размечены всевозможными SLD-резольвентами запроса $\mathfrak{C}(v)$ с утверждениями $\mathcal{P}$

(это формулируется для
стандартного правила выбора подцели)

Деревья вычислений

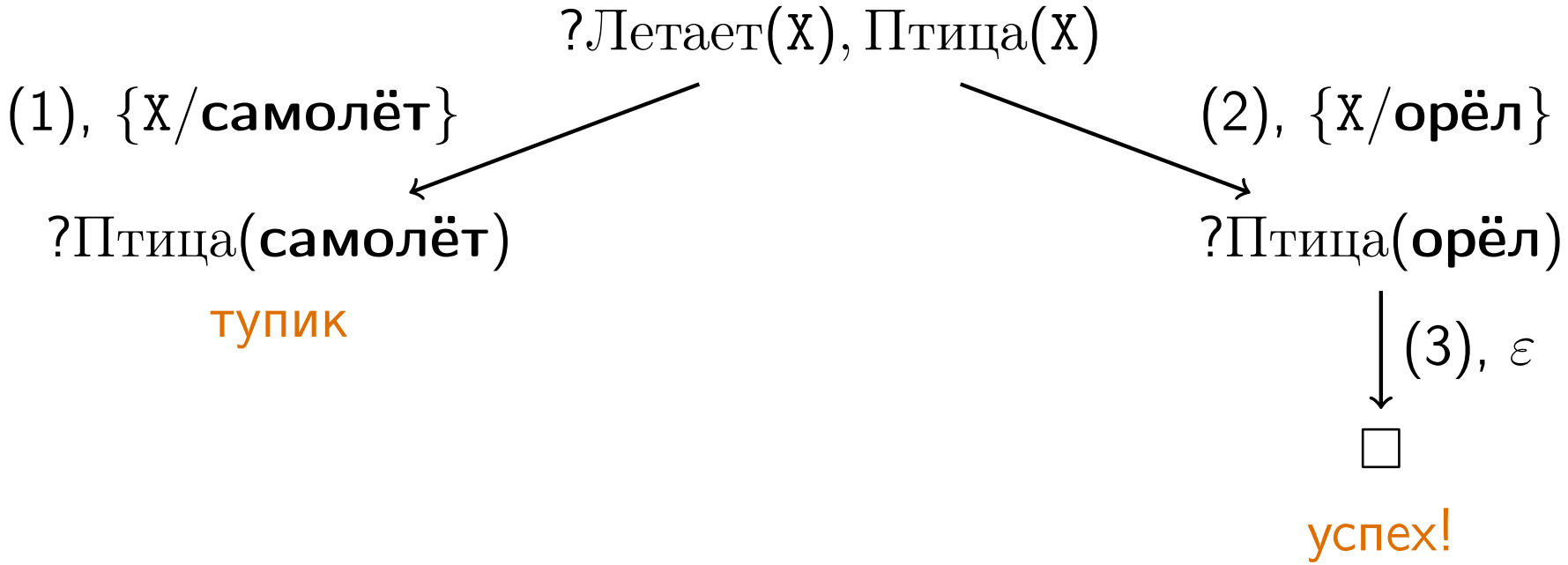
Иллюстрация определения дерева вычислений:



Деревья вычислений

Пример

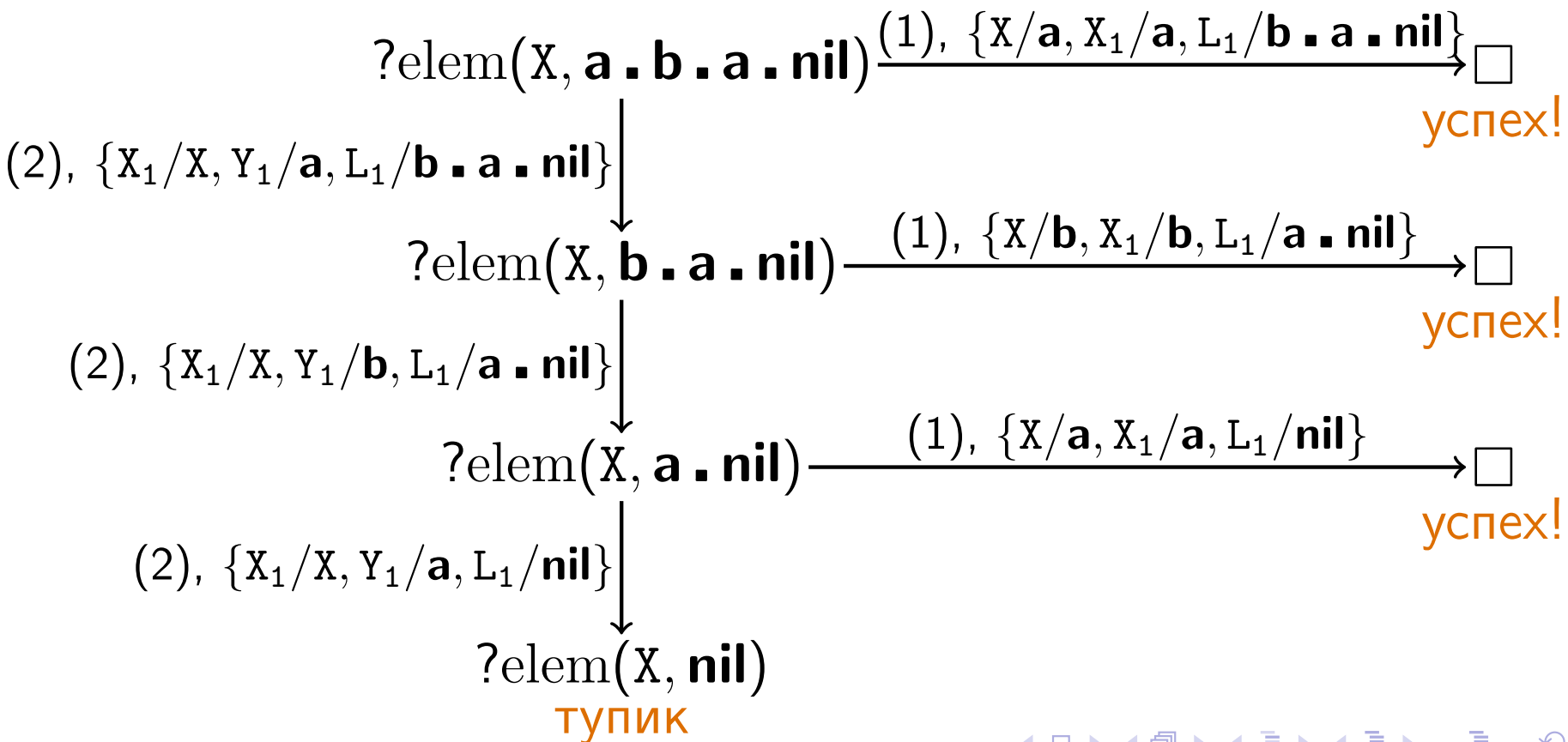
(1) Летает(**самолёт**);
(2) Летает(**орёл**);
(3) Птица(**орёл**);
(4) Птица(**пингвин**);



Деревья вычислений

Пример

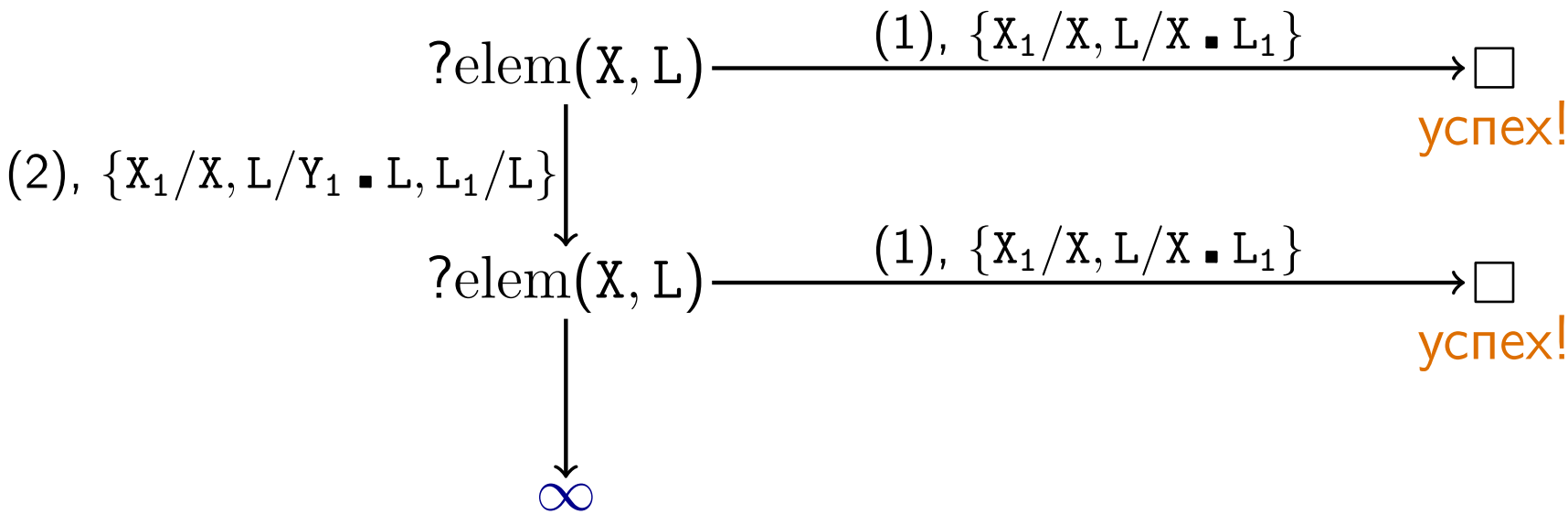
$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \mathbin{.} \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \mathbin{.} \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Деревья вычислений

Пример

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Деревья вычислений

Деревья вычислений бывают разные: ни одного успеха, ровно один успех, много успехов, линейные, конечные, бесконечные, ...

SLD-резолютивные вычисления **в точности соответствуют** всем ветвям дерева вычислений

Некоторые ветви успешные и позволяют получить вычисленный ответ

Некоторые ветви тупиковые и к ответу не приводят

Некоторые ветви бесконечные, что ещё хуже

И вопрос о переборе утверждений можно записать так:

как обнаружить **все** успешные ветви дерева вычислений?

Правила поиска

Правило поиска определяет, как следует обходить дерево SLD-резолютивных вычислений для любого запроса $\mathfrak{C}$ к любой программе $\mathcal{P}$

Как может выглядеть правило поиска?

Многие известные алгоритмы обхода графов могут служить в роли правил поиска, например:

- обход в ширину:
  - дерево строится (или обходится) поярусно: как только построены все вершины i-го яруса, начинаются строиться вершины $(i+1)$-го яруса
- обход в глубину с возвратом
  - одна ветвь дерева обходится до конца
  - затем происходит возврат (или откат) до вершины, из которой исходит следующая ветвь, и она обходится до конца
  - и так поочерёдно обходятся все ветви

Правила поиска

Пример обхода в ширину

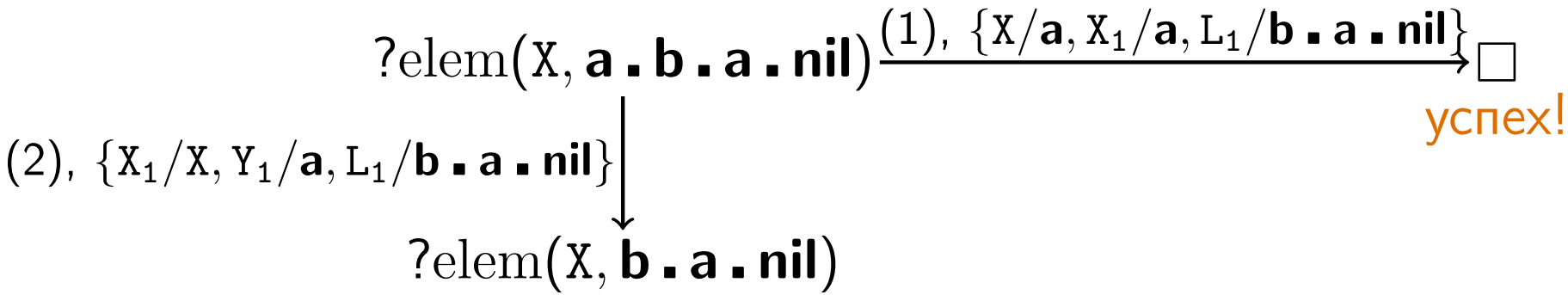
$$(1) \quad \mathrm{elem}(\mathtt{X}, \mathtt{X}\,\mathbf{.}\,\mathtt{L});$$
$$(2) \quad \mathrm{elem}(\mathtt{X}, \mathtt{Y}\,\mathbf{.}\,\mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

$$?\mathrm{elem}(\mathtt{X}, \mathbf{a}\,\mathbf{.}\,\mathbf{b}\,\mathbf{.}\,\mathbf{a}\,\mathbf{.}\,\mathbf{nil})$$

Правила поиска

Пример обхода в ширину

$$(1) \quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2) \quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Правила поиска

Пример обхода в ширину

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

$$?\mathrm{elem}(\mathtt{X}, \mathbf{a} \,.\, \mathbf{b} \,.\, \mathbf{a} \,.\, \mathbf{nil}) \xrightarrow{(1),\ \{\mathtt{X}/\mathbf{a}, \mathtt{X}_1/\mathbf{a}, \mathtt{L}_1/\mathbf{b} \,.\, \mathbf{a} \,.\, \mathbf{nil}\}} \square$$

успех!

$$\downarrow\ (2),\ \{\mathtt{X}_1/\mathtt{X}, \mathtt{Y}_1/\mathbf{a}, \mathtt{L}_1/\mathbf{b} \,.\, \mathbf{a} \,.\, \mathbf{nil}\}$$

$$?\mathrm{elem}(\mathtt{X}, \mathbf{b} \,.\, \mathbf{a} \,.\, \mathbf{nil}) \xrightarrow{(1),\ \{\mathtt{X}/\mathbf{b}, \mathtt{X}_1/\mathbf{b}, \mathtt{L}_1/\mathbf{a} \,.\, \mathbf{nil}\}} \square$$

успех!

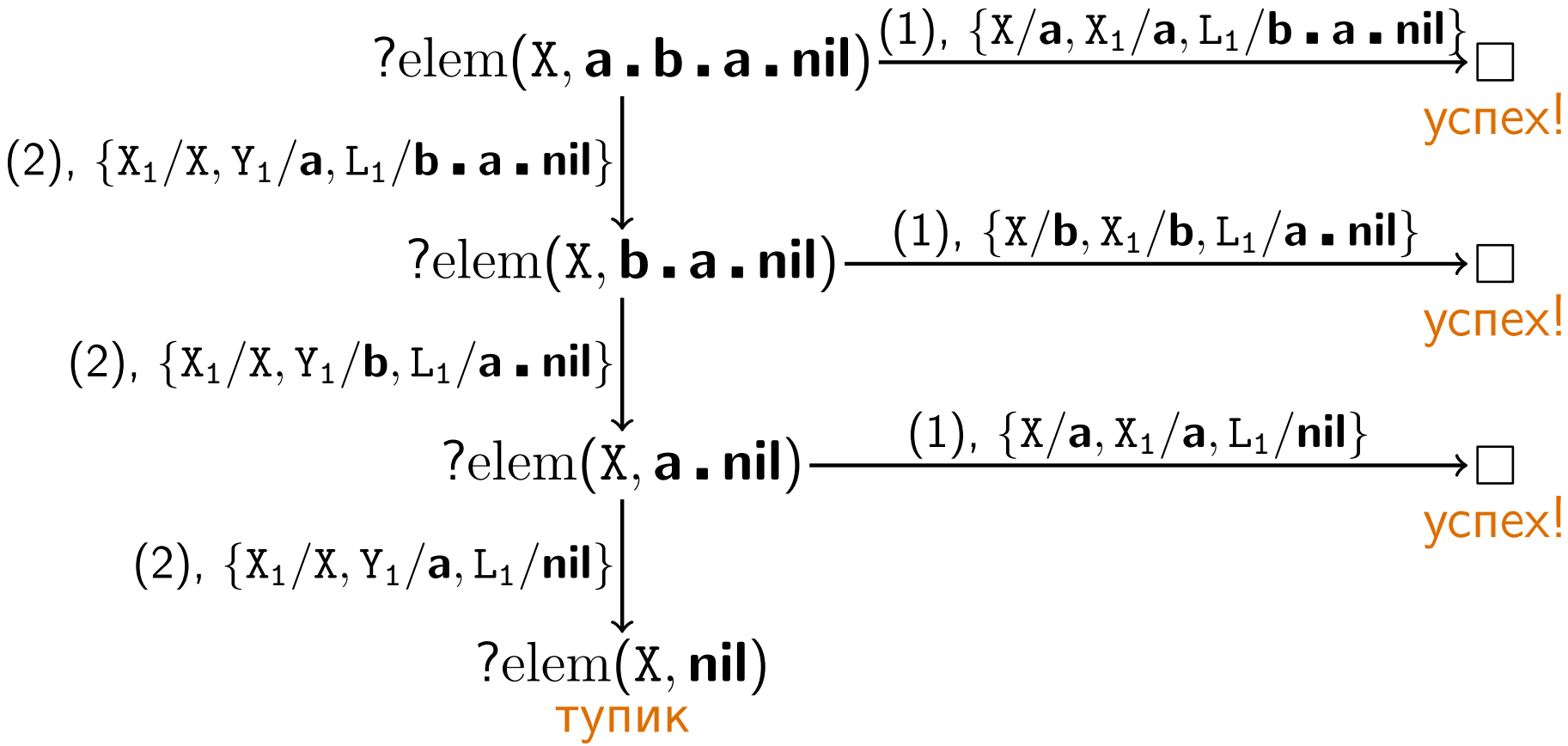
$$\downarrow\ (2),\ \{\mathtt{X}_1/\mathtt{X}, \mathtt{Y}_1/\mathbf{b}, \mathtt{L}_1/\mathbf{a} \,.\, \mathbf{nil}\}$$

$$?\mathrm{elem}(\mathtt{X}, \mathbf{a} \,.\, \mathbf{nil})$$

Правила поиска

Пример обхода в ширину

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \mathbin{.} \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \mathbin{.} \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Правила поиска

Пример обхода в ширину

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \mathbf{.} \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \mathbf{.} \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

СТОП: дерево построено

Правила поиска

Пример обхода в глубину с возвратом

$$(1) \quad \mathrm{elem}(\mathtt{X}, \mathtt{X}\,\mathbf{.}\,\mathtt{L});$$
$$(2) \quad \mathrm{elem}(\mathtt{X}, \mathtt{Y}\,\mathbf{.}\,\mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

$$?\mathrm{elem}(\mathtt{X}, \mathbf{a}\,\mathbf{.}\,\mathbf{b}\,\mathbf{.}\,\mathbf{a}\,\mathbf{.}\,\mathbf{nil})$$

Правила поиска

Пример обхода в глубину с возвратом

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

$$?\mathrm{elem}(\mathtt{X}, \mathbf{a} \,.\, \mathbf{b} \,.\, \mathbf{a} \,.\, \mathbf{nil}) \xrightarrow{(1),\ \{\mathtt{X}/\mathbf{a}, \mathtt{X}_1/\mathbf{a}, \mathtt{L}_1/\mathbf{b} \,.\, \mathbf{a} \,.\, \mathbf{nil}\}} \square$$

Правила поиска

Пример обхода в глубину с возвратом

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X}\,\mathbf{.}\,\mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y}\,\mathbf{.}\,\mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

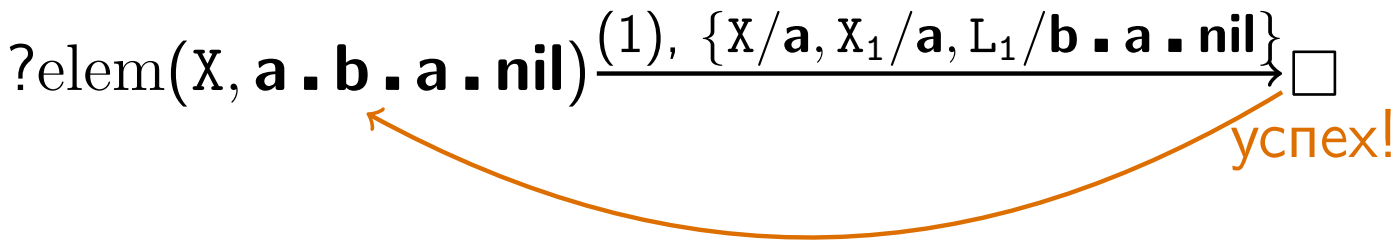
$$?\mathrm{elem}(\mathtt{X}, \mathbf{a}\,\mathbf{.}\,\mathbf{b}\,\mathbf{.}\,\mathbf{a}\,\mathbf{.}\,\mathbf{nil}) \xrightarrow{(1),\ \{\mathtt{X}/\mathbf{a}, \mathtt{X}_1/\mathbf{a}, \mathtt{L}_1/\mathbf{b}\,\mathbf{.}\,\mathbf{a}\,\mathbf{.}\,\mathbf{nil}\}} \square$$

успех!

Правила поиска

Пример обхода в глубину с возвратом

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

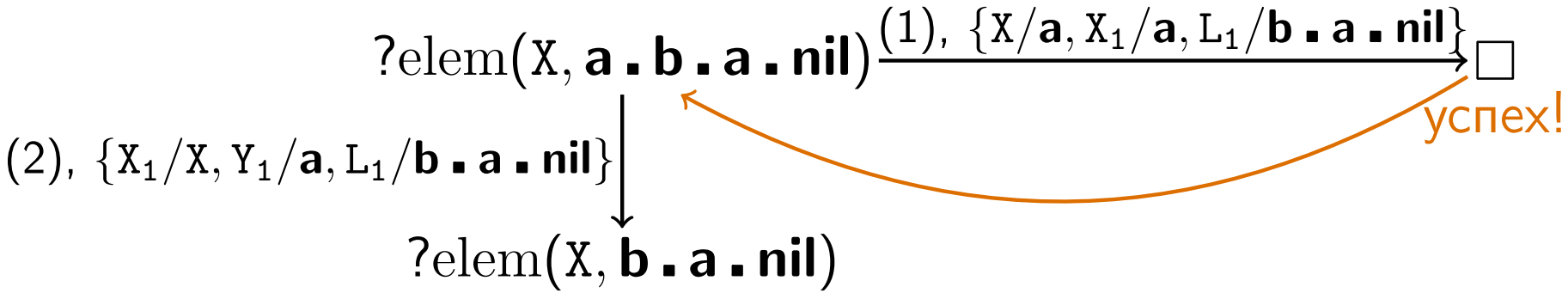


Правила поиска

Пример обхода в глубину с возвратом

(1) $\mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L})$;

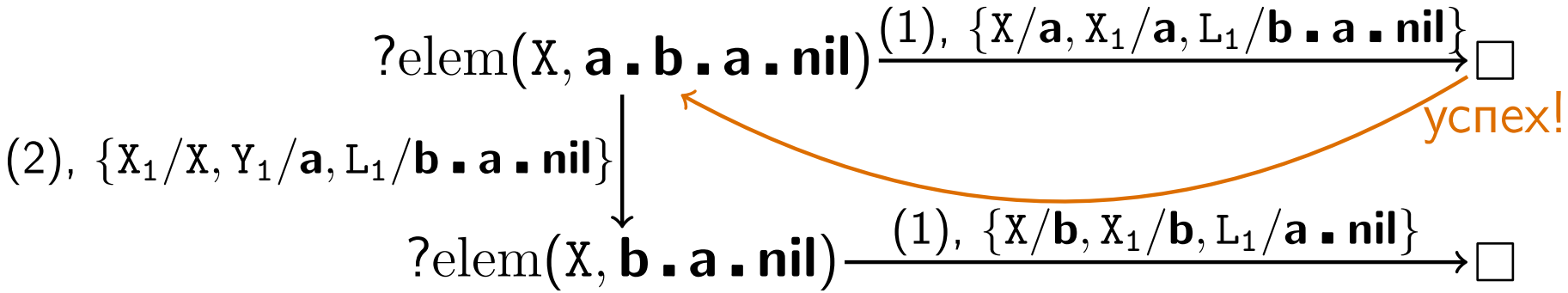
(2) $\mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;



Правила поиска

Пример обхода в глубину с возвратом

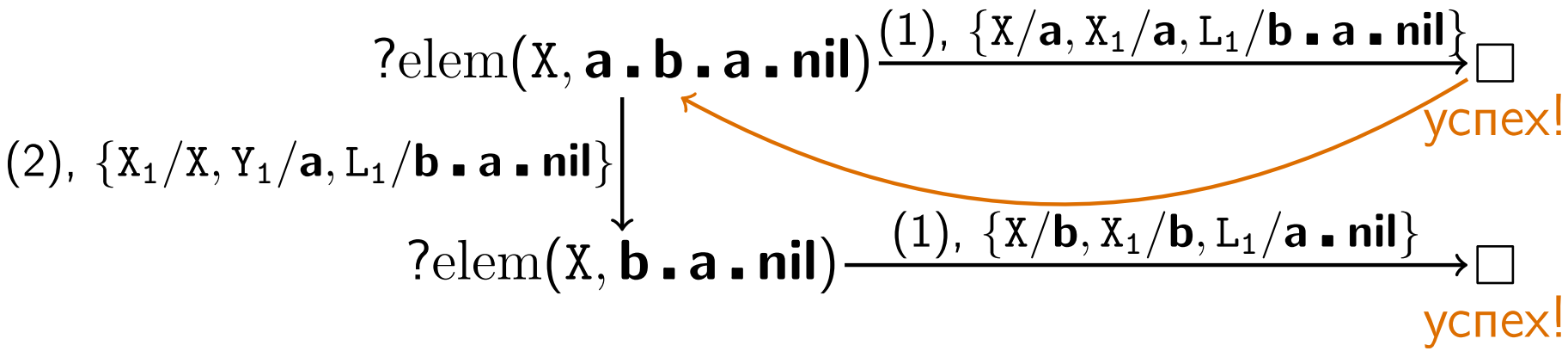
$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Правила поиска

Пример обхода в глубину с возвратом

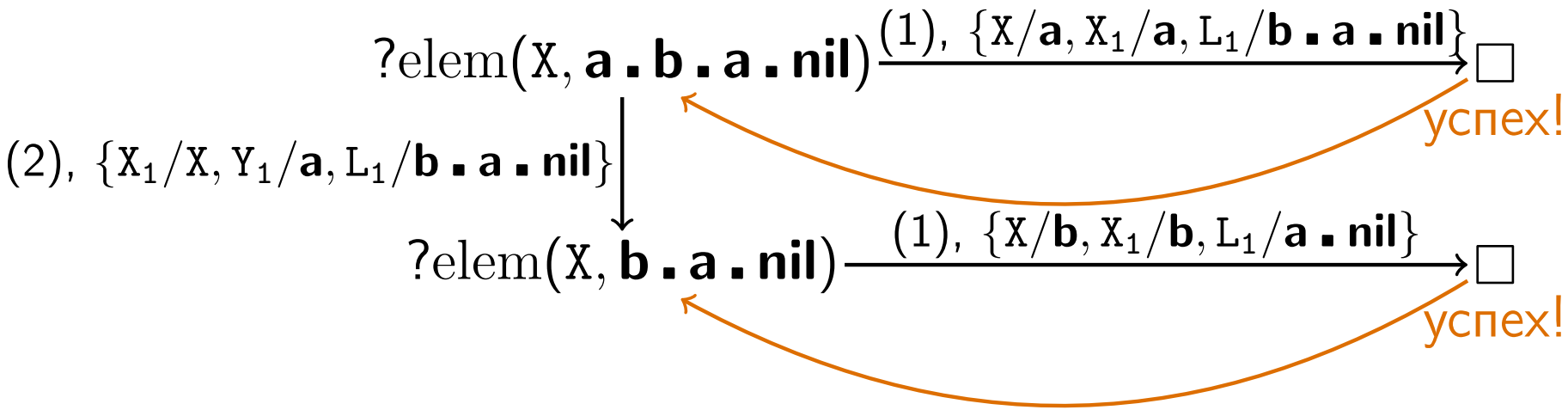
(1) $\mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L})$;
(2) $\mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;



Правила поиска

Пример обхода в глубину с возвратом

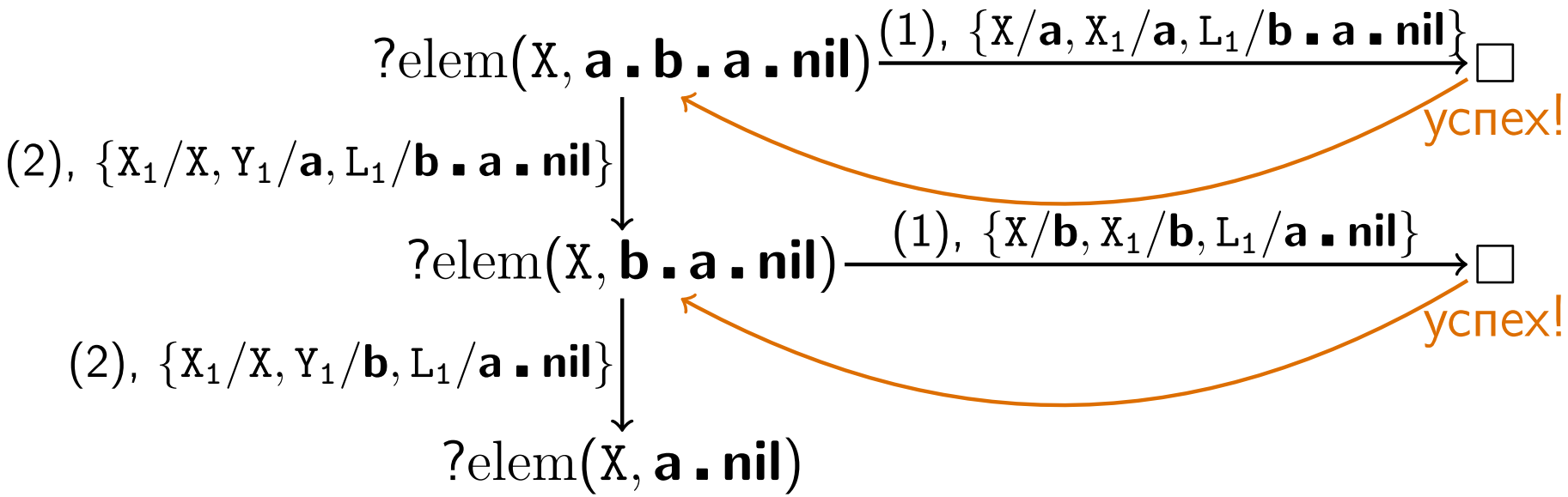
(1) $\mathrm{elem}(\mathtt{X}, \mathtt{X} \mathbf{.} \mathtt{L})$;
(2) $\mathrm{elem}(\mathtt{X}, \mathtt{Y} \mathbf{.} \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;



Правила поиска

Пример обхода в глубину с возвратом

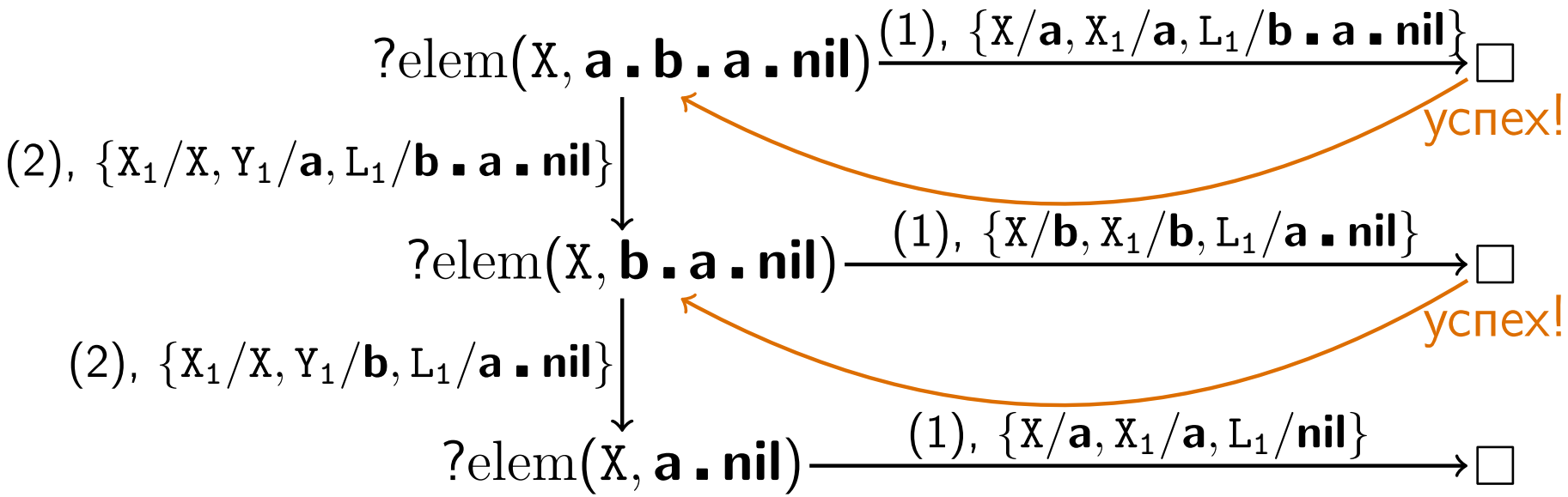
$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Правила поиска

Пример обхода в глубину с возвратом

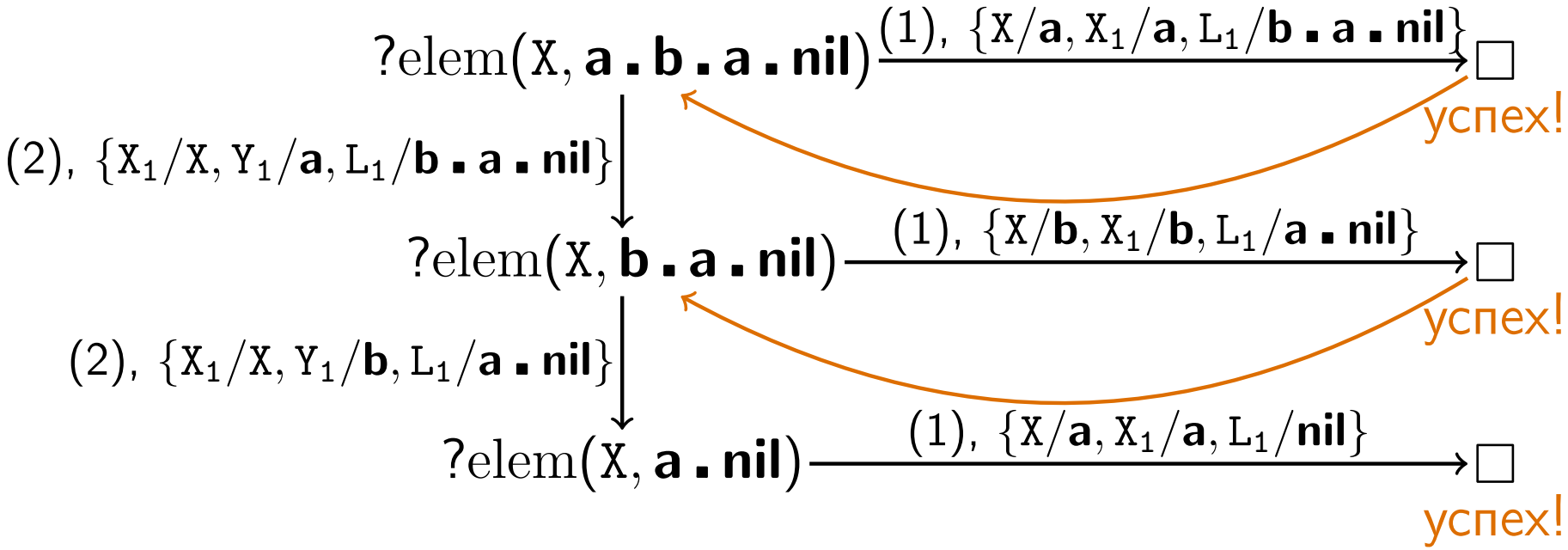
$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Правила поиска

Пример обхода в глубину с возвратом

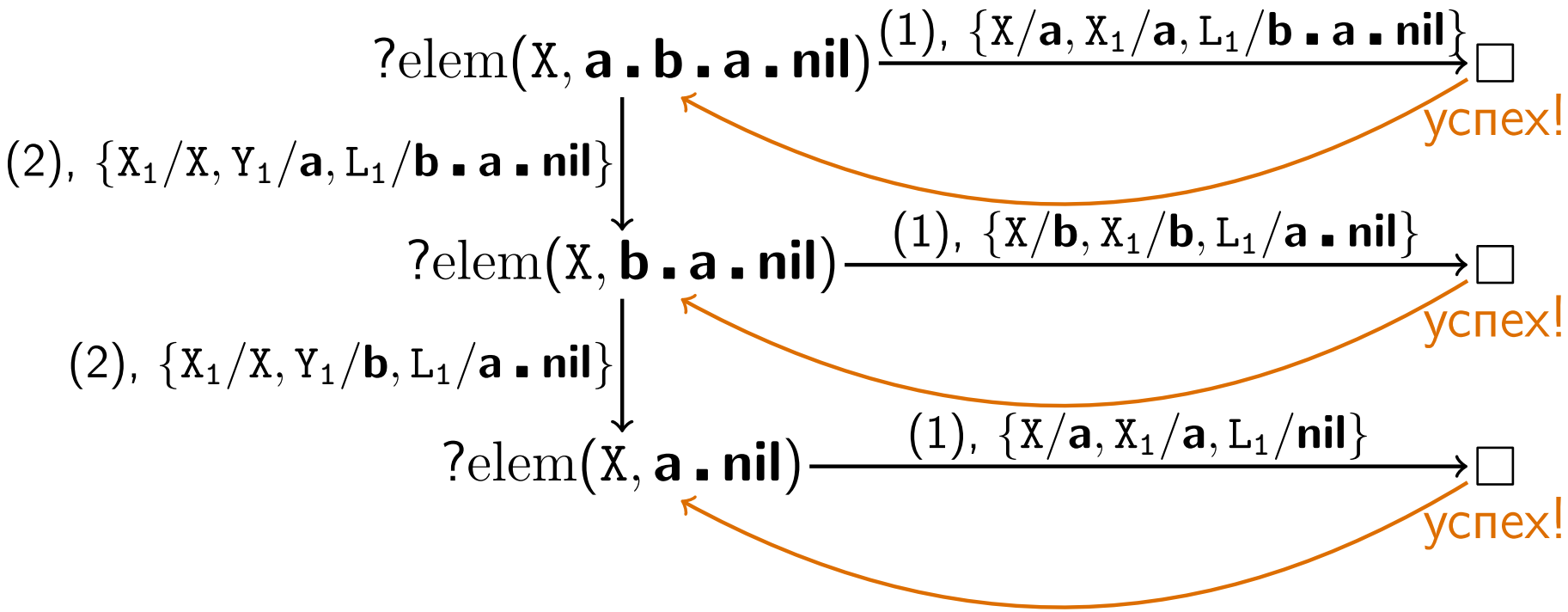
$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Правила поиска

Пример обхода в глубину с возвратом

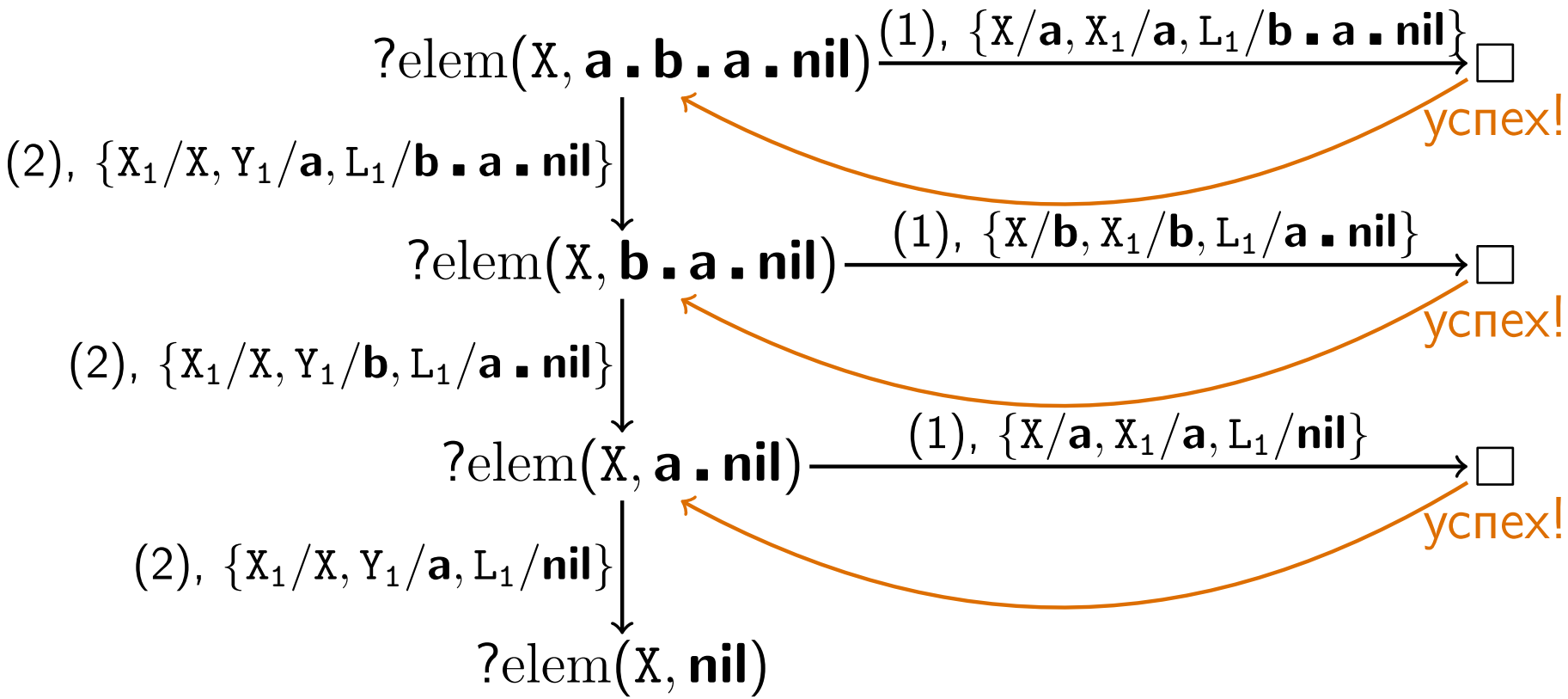
$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Правила поиска

Пример обхода в глубину с возвратом

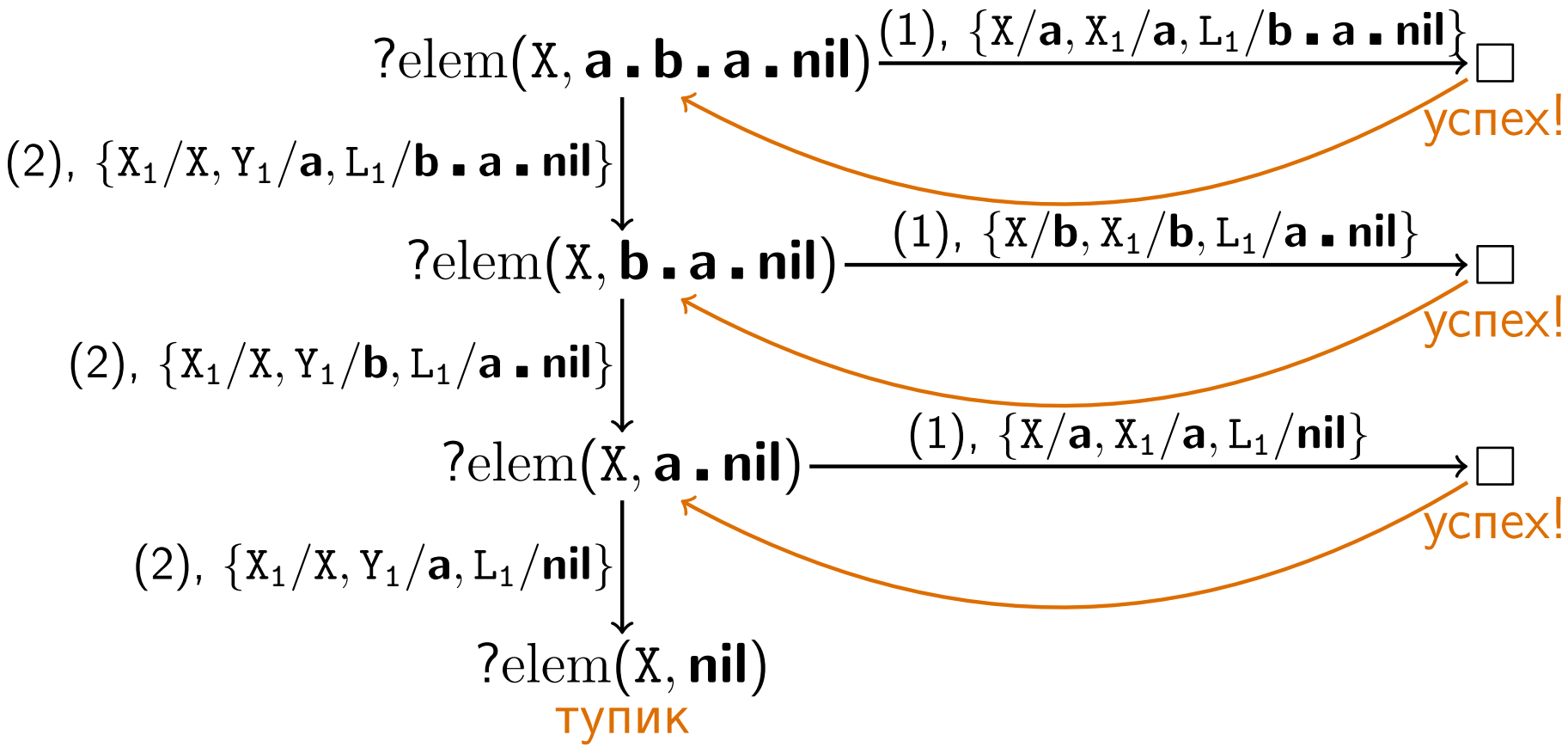
(1) $\mathrm{elem}(\mathtt{X}, \mathtt{X} \mathbin{.} \mathtt{L})$;
(2) $\mathrm{elem}(\mathtt{X}, \mathtt{Y} \mathbin{.} \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;



Правила поиска

Пример обхода в глубину с возвратом

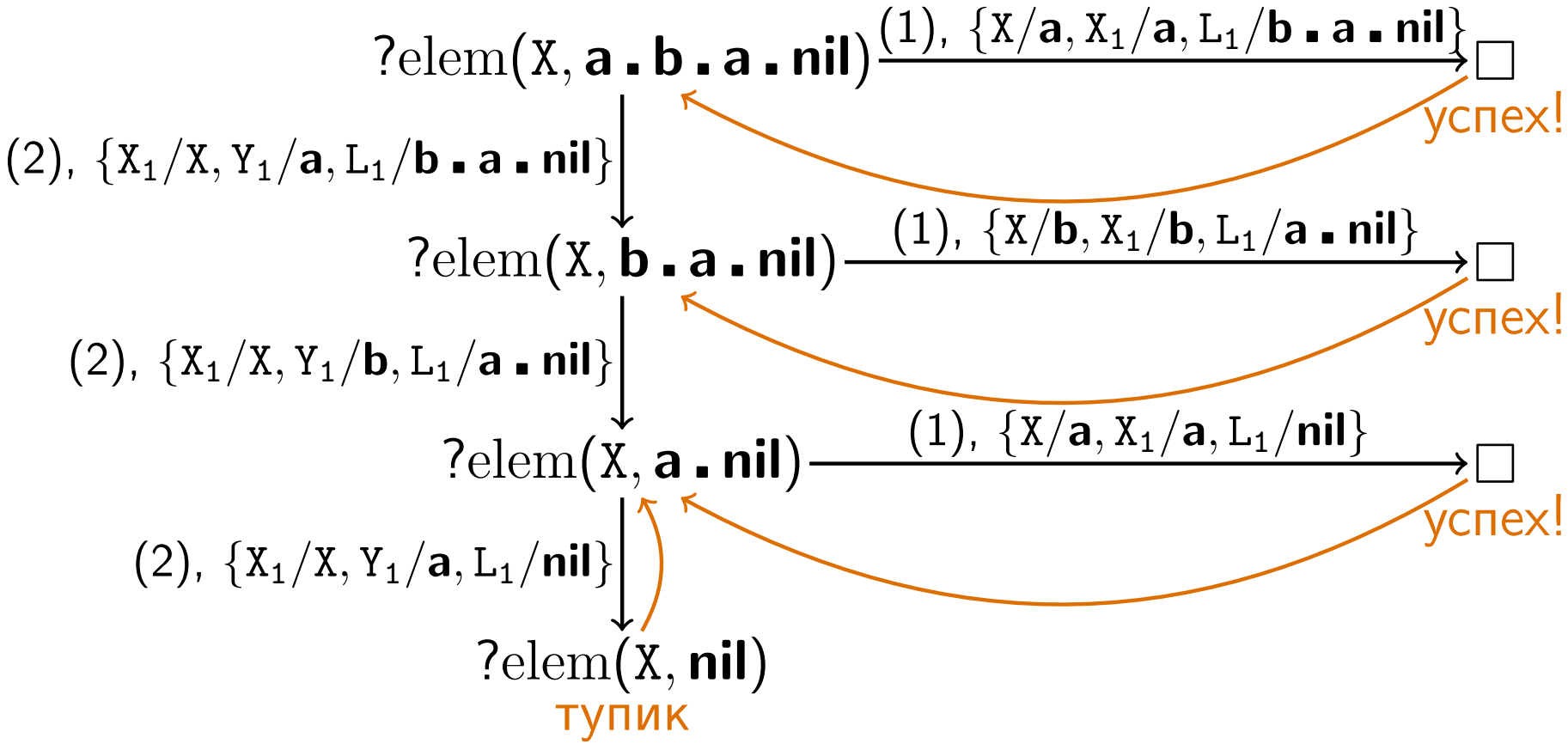
(1) $\mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L})$;
(2) $\mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;



Правила поиска

Пример обхода в глубину с возвратом

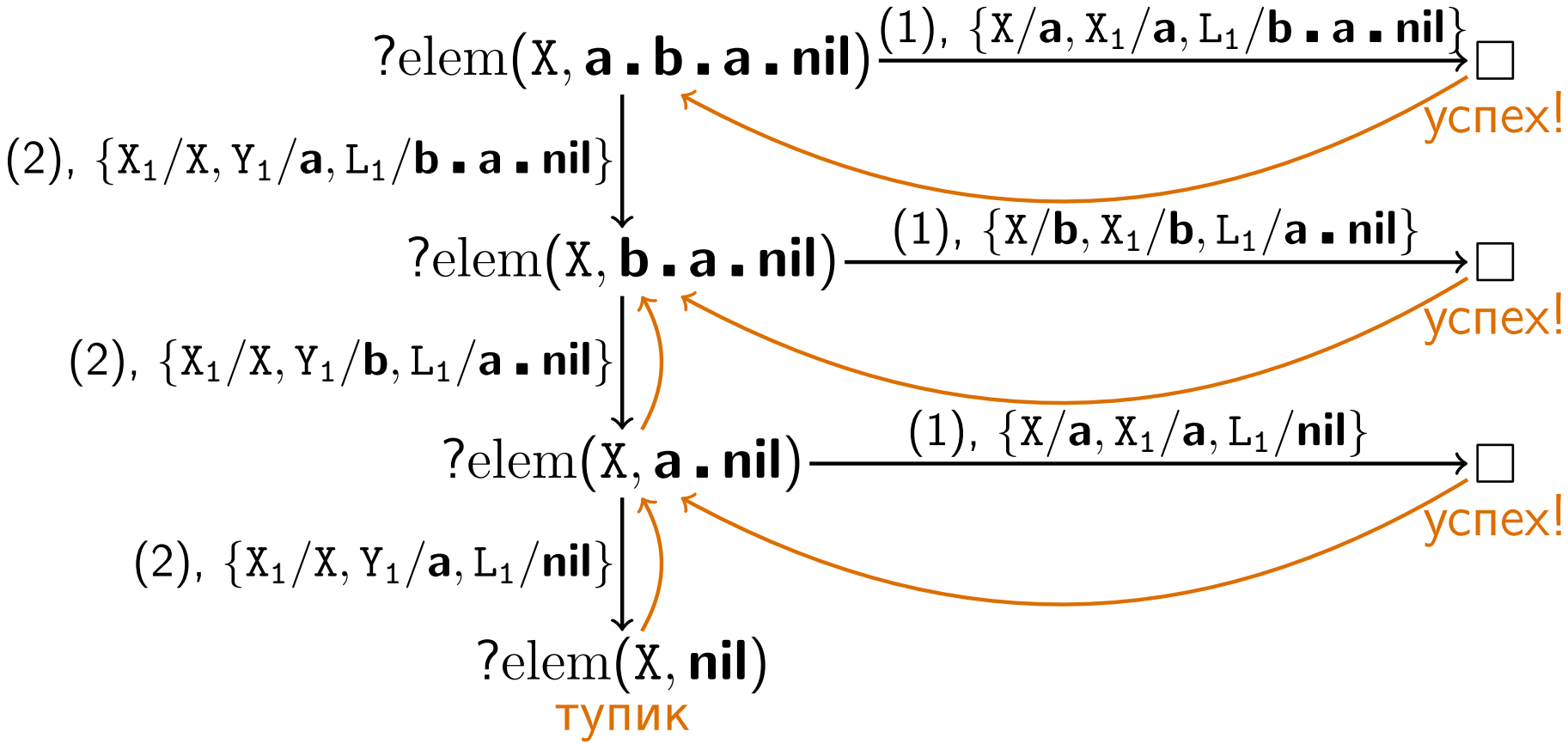
$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Правила поиска

Пример обхода в глубину с возвратом

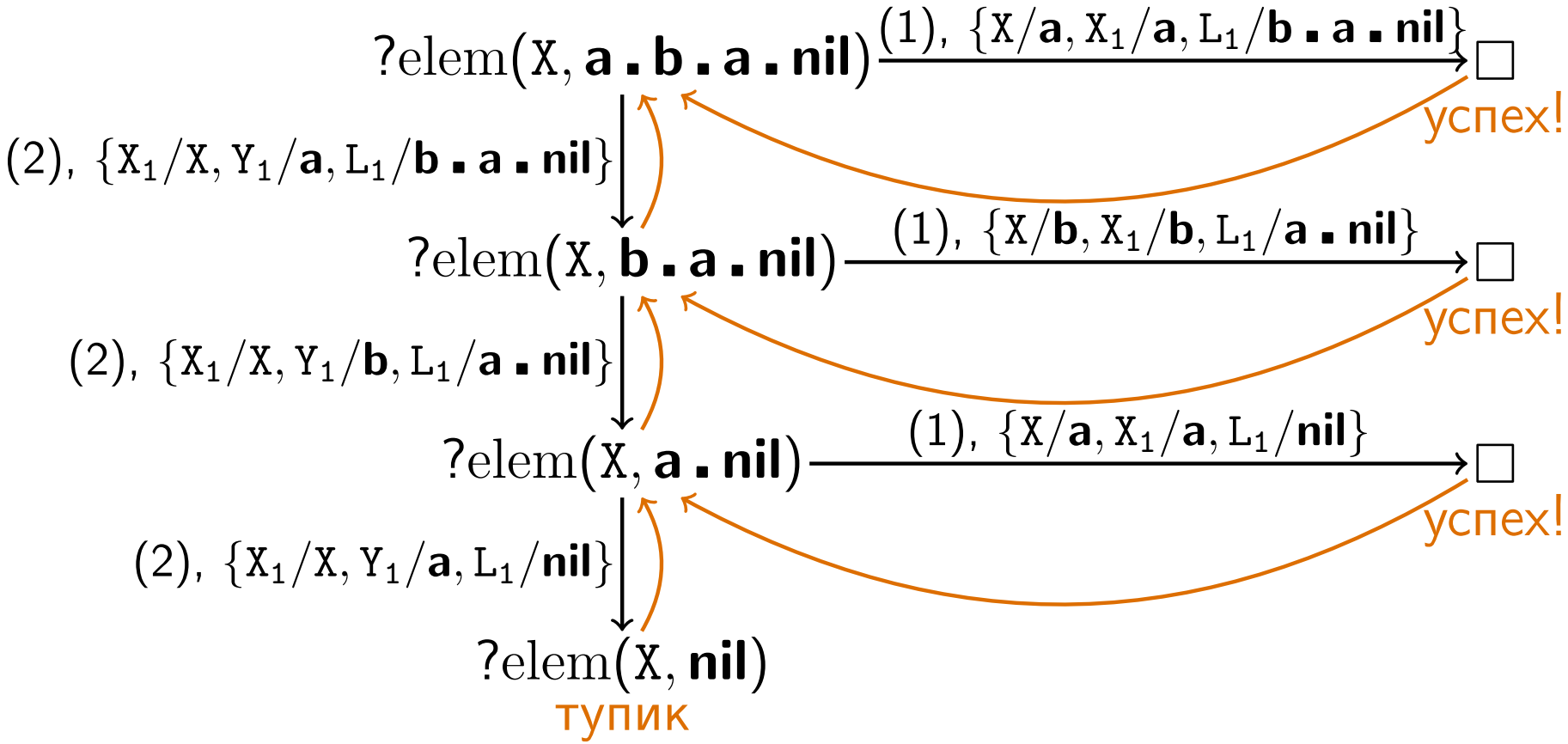
$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



Правила поиска

Пример обхода в глубину с возвратом

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

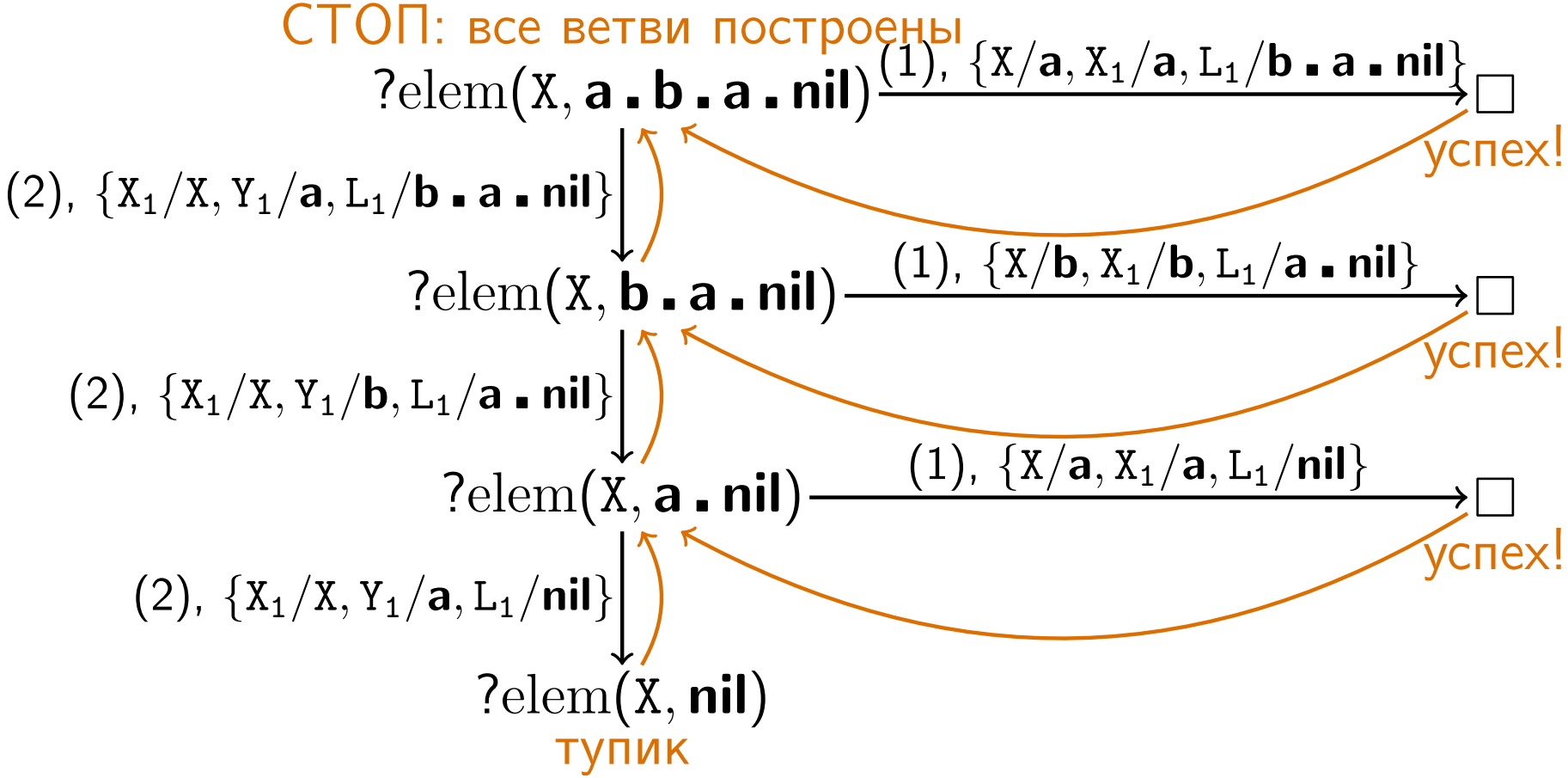


Правила поиска

Пример обхода в глубину с возвратом

(1) $\mathrm{elem}(\mathtt{X}, \mathtt{X} \mathbf{.} \mathtt{L})$;

(2) $\mathrm{elem}(\mathtt{X}, \mathtt{Y} \mathbf{.} \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;



Стратегии вычисления

Правило выбора подцели и правило поиска в совокупности образуют стратегию вычисления запросов к хорновским логическим программам

Стратегия вычисления называется вычислительно полной, если для любого запроса $\mathfrak{C}$ к любой программе $\mathcal{P}$ она строит **все** успешные вычисления, то есть позволяет найти **все** вычисленные ответы на запрос $\mathfrak{C}$ к программе $\mathcal{P}$

Остановимся на стандартном правиле выбора подцели

В каком случае (т.е. для какого правила поиска) стратегия вычисления будет вычислительно полной?

Стратегии вычисления

Стратегия, основанная на обходе в ширину, **является** вычислительно полной:

- любой запрос к любой (конечной) программе имеет лишь конечное число SLD-резольвент, то есть число вершин каждого яруса конечно
- любое успешное вычисление имеет конечную длину, то есть оканчивается на некотором ярусе
- значит, любое успешное вычисление будет рано или поздно построено

Но такая стратегия **не используется** в интерпретаторах:

- пусть для каждого запроса в дереве вычислений существует ровно две SLD-резольвенты
- тогда на N-м ярусе будет 2^N вершин
- и все вершины одного яруса нужно хранить в памяти
- итого достаточно **350** ярусов, чтобы получить гугол вершин

(а это очень большое число: см. лекцию 5)

Стратегии вычисления

С другой стороны, стратегия, основанная на обходе в глубину с возвратом,

- имеет эффективную реализацию и реально применяется: достаточно
  - пронумеровать утверждения программы в порядке, в котором они встречаются в тексте
  - при выборе ветви просматривать правила по порядку и выбирать первое подходящее для построения SLD-резольвенты
  - хранить в памяти лишь те запросы, которые относятся к текущей строящейся ветви
- но при этом, к сожалению, она является **вычислительно неполной**

Стратегии вычисления

Например,

$$
\begin{array}{ll}
(1) & \mathrm{elem}(\mathtt{X}, \mathtt{Y}\,.\,\mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L}); \\
(2) & \mathrm{elem}(\mathtt{X}, \mathtt{X}\,.\,\mathtt{L});
\end{array}
$$

$$
?\mathrm{elem}(\mathtt{X}, \mathtt{L})
$$

Стратегии вычисления

Например,

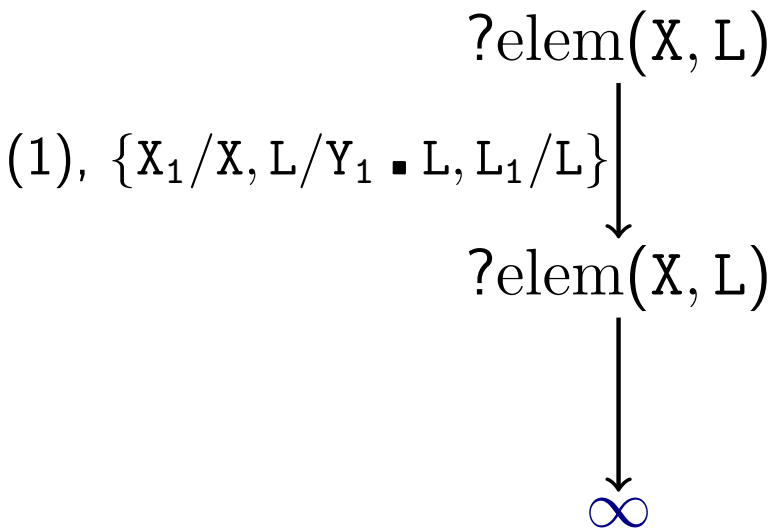
$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \centerdot \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \centerdot \mathtt{L});$$

$$?\mathrm{elem}(\mathtt{X}, \mathtt{L})$$
$$\Big\downarrow (1),\ \{\mathtt{X}_1/\mathtt{X}, \mathtt{L}/\mathtt{Y}_1 \centerdot \mathtt{L}, \mathtt{L}_1/\mathtt{L}\}$$
$$?\mathrm{elem}(\mathtt{X}, \mathtt{L})$$

Стратегии вычисления

Например,

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$

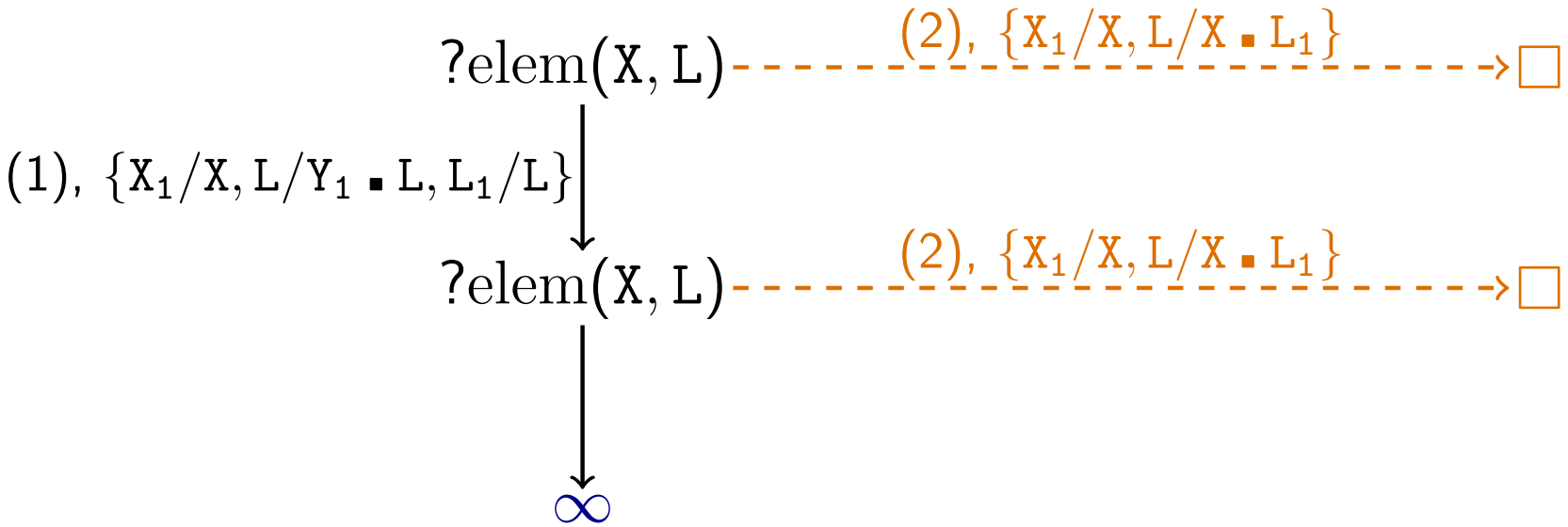


- Строим бесконечную ветвь и не выдаём ни одного ответа

Стратегии вычисления

Например,

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$

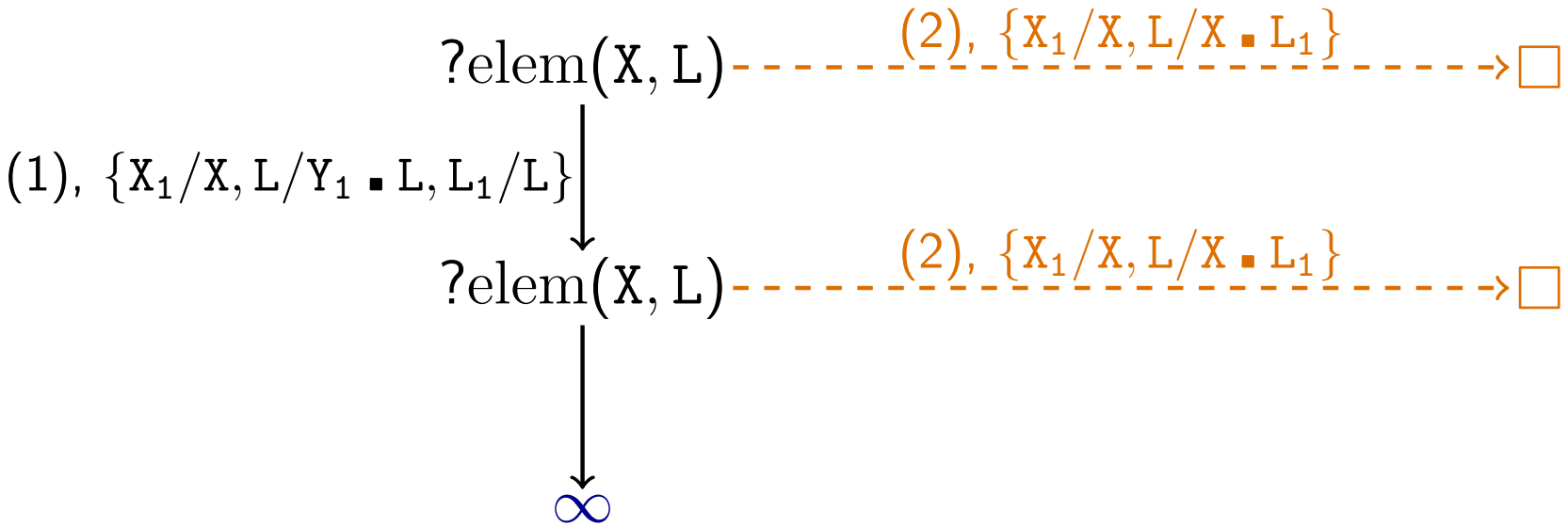


- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много

Стратегии вычисления

Например,

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$



- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

Например,

$$(1) \quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \,.\, \mathtt{L});$$
$$(2) \quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \,.\, \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

$$?\mathrm{elem}(\mathtt{X}, \mathtt{L})$$

- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

Например,

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

$$?\mathrm{elem}(\mathtt{X}, \mathtt{L}) \xrightarrow{(2),\ \{\mathtt{X}_1/\mathtt{X}, \mathtt{L}/\mathtt{X} \blacksquare \mathtt{L}_1\}} \square$$

- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

Например,

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

$$?\mathrm{elem}(\mathtt{X}, \mathtt{L}) \xrightarrow{(2),\ \{\mathtt{X}_1/\mathtt{X}, \mathtt{L}/\mathtt{X} \blacksquare \mathtt{L}_1\}} \square$$

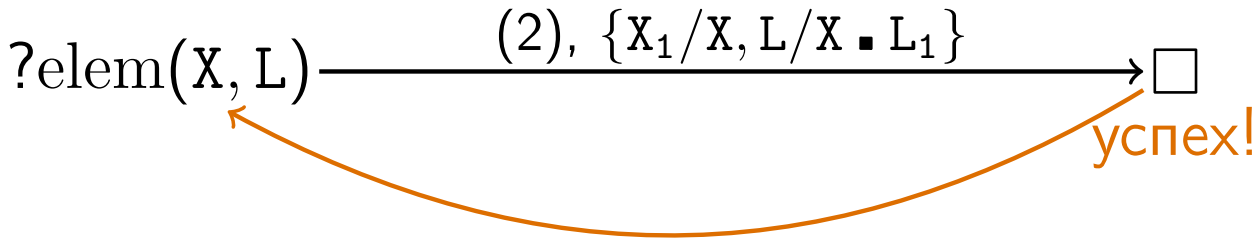
успех!

- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

Например,

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

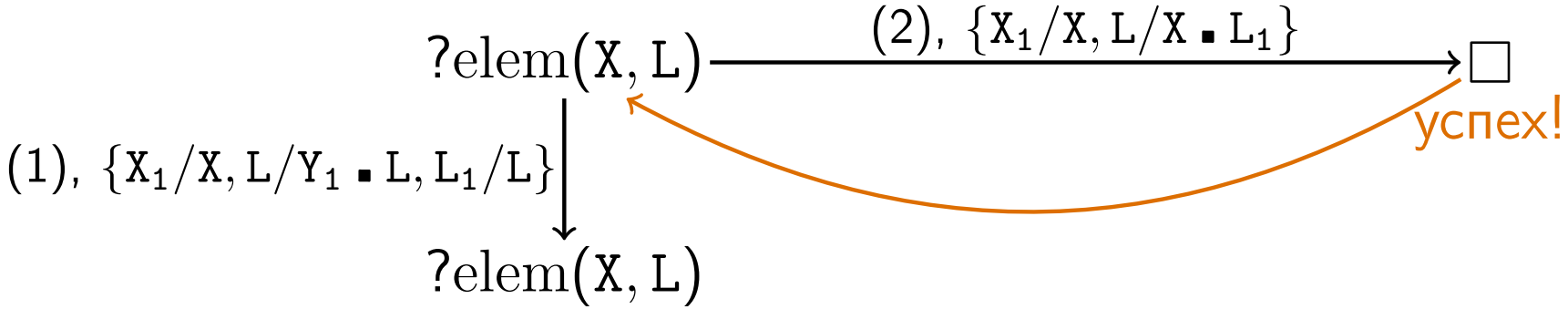


- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

Например,

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

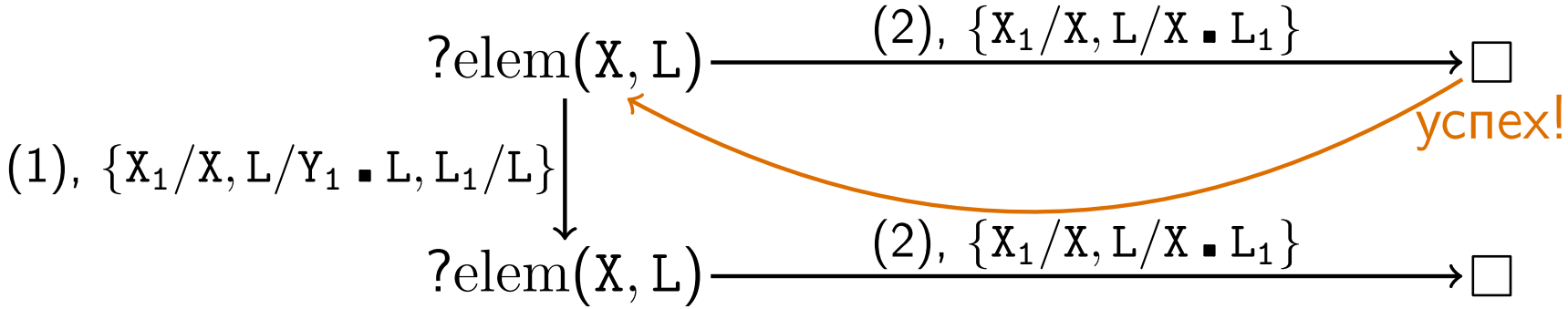


- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

Например,

(1) $\mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L})$;
(2) $\mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;

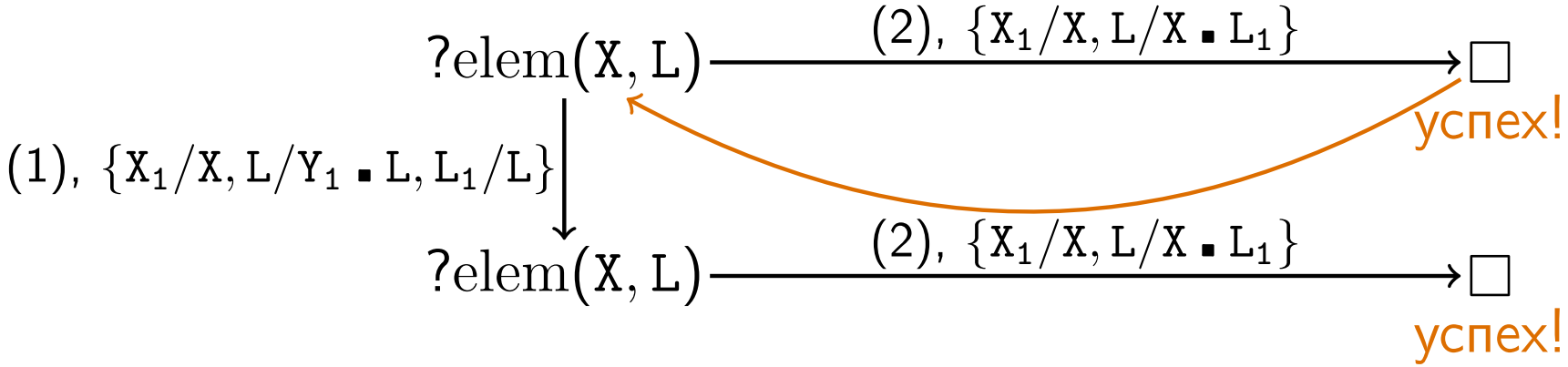


- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

Например,

(1) $\mathrm{elem}(\mathtt{X}, \mathtt{X} \centerdot \mathtt{L})$;
(2) $\mathrm{elem}(\mathtt{X}, \mathtt{Y} \centerdot \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;

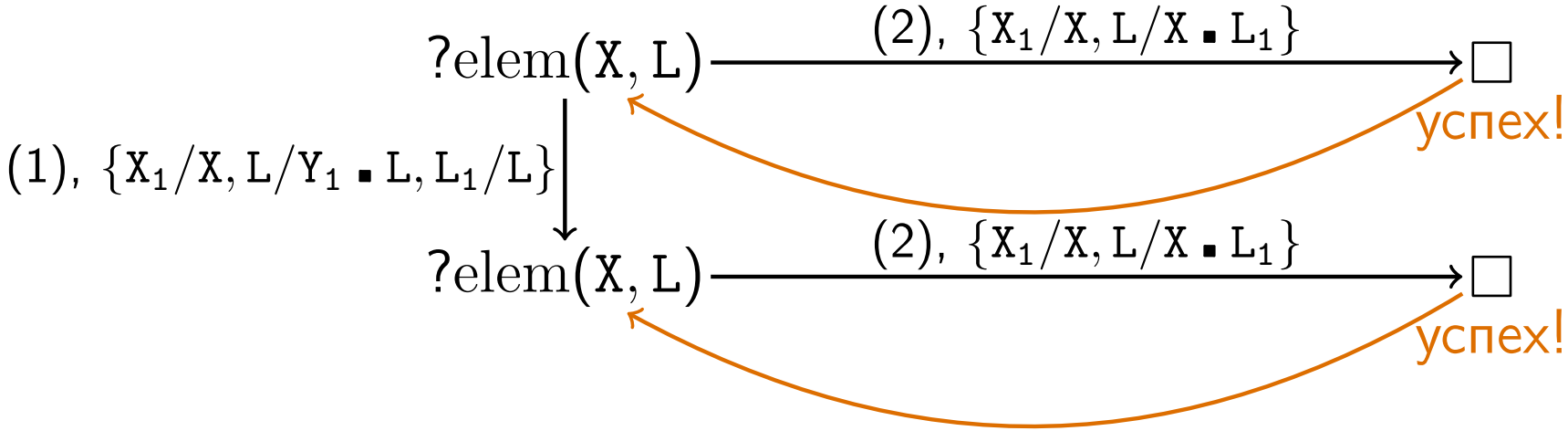


- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

Например,

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$

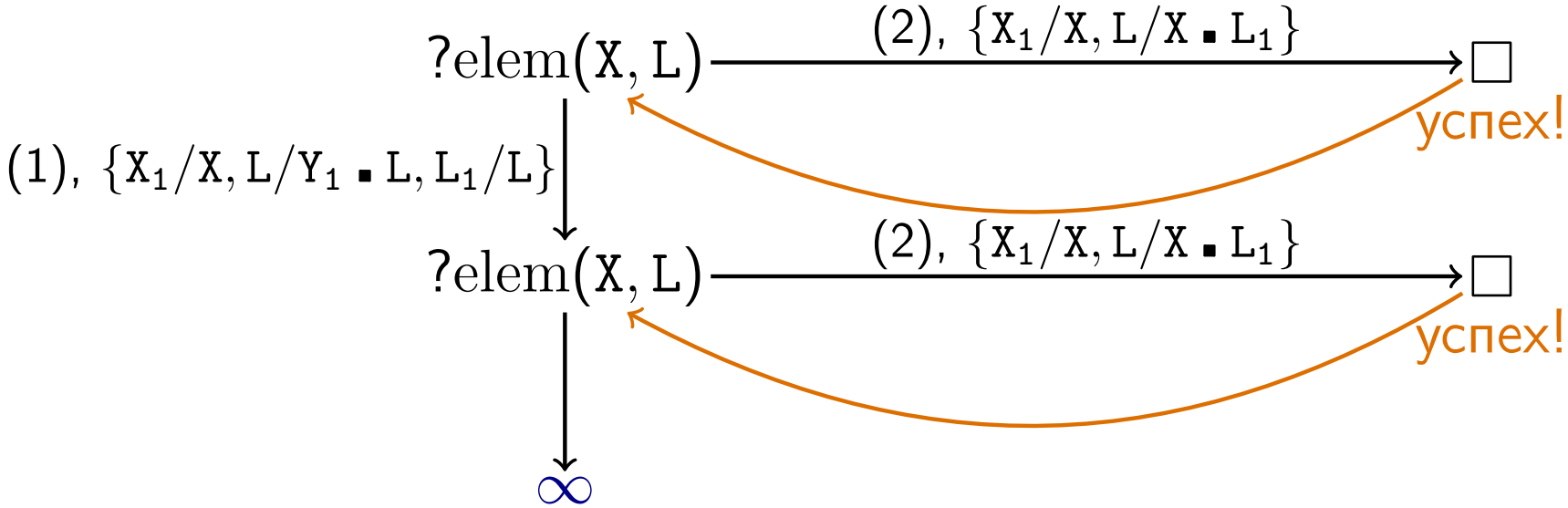


- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

Например,

$$(1)\quad \mathrm{elem}(\mathtt{X}, \mathtt{X} \blacksquare \mathtt{L});$$
$$(2)\quad \mathrm{elem}(\mathtt{X}, \mathtt{Y} \blacksquare \mathtt{L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$



- Строим бесконечную ветвь и не выдаём ни одного ответа
- При этом вычисленных ответов бесконечно много
- Более того, эта стратегия оказывается чувствительной к порядку правил

Стратегии вычисления

В программировании эффективность намного важнее полноты:

> если что-то работает **неразумно долго**,
> то считается, что оно и **не работает вовсе**

Поэтому в стандартную стратегию вычисления, которая обычно реализована в интерпретаторах, входят

- стандартное правило выбора подцели: всегда обрабатывай самую левую подцель — и
- обход дерева SLD-резолютивных вычислений в глубину с возвратом с применением утверждений по порядку их вхождения в текст программы

А расставлять утверждения в надлежащем порядке программист обязан **сам**

Конец лекции 14