Математическая логика и логическое программирование

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

Блок 1

Вступление:
- Что такое логика
- Несколько логических парадоксов
- Чего ожидать в лекциях

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

ВМК МГУ, 2023/2024, осенний семестр

Что такое логика

Логика[1] — это наука об общезначимых формах и средствах мысли, необходимых для рационального познания в любой области знания

Логика[2] — это нормативная наука о формах и приёмах интеллектуальной познавательной деятельности, осуществляемой с помощью языка

Логика[3] — это наука, которая изучает, каким образом мы выражаем мысли, делаем умозаключения, и как всё это можно представить формально

Логика[4] — это наука о правильных способах рассуждения, то есть таких, при которых из верных исходных положений получаются верные результаты

---

1 Философский энциклопедический словарь.
2 Бочаров, Маркин. Основы логики.
3 Метакидес, Нероуд. Принципы логики и логического программирования.
4 Успенский, Верещагин, Плиско. Вводный курс математической логики.

Что такое логика

Формальная логика[1] — это наука о законах знания,
полученного из ранее установленных и проверенных истин,
без обращения в каждом конкретном случае к опыту,
а только в результате применения законов и правил мышления

Формальная логика[2] — сейчас это синоним логики

Математическая логика[3] — это
1) логика, развиваемая с помощью математических методов,
2) логика, используемая в математике

1 Кондаков. Логический словарь-справочник.
2 Философский энциклопедический словарь.
3 Клини. Математическая логика.

Что такое логика

Формальная математическая логика изучает

- законы причинно-следственной связи в окружающем мире
- проявление этих законов в рациональном мышлении человека
- отражение этих законов в языках,
  - как естественных,
  - так и искусственных
- формы, в которых проявляются эти законы,
  вне зависимости от содержания (смысла)
  тех явлений (предметов), к которым эти законы относятся

Что такое логика

Например:

Из того, что
все люди смертны
и
Сократ — человек
следует, что
Сократ смертен

Из того, что
у змей нет лап
и
уж — змея
следует, что
у ужа нет лап

Из того, что
каждый металл — проводник
и
ртуть — металл
следует, что
ртуть — проводник

Эти высказывания относятся к совершенно разным областям
и имеют совершенно разный смысл, но в целом одинаковую форму

Что такое логика

Например:

Вот эта форма:

Из того, что
каждый предмет, обладающий свойством A
обладает и свойством B
и
предмет c обладает свойством A
следует, что
предмет c обладает свойством B

Это один из законов логики, не зависящий от смысла A, B и **c**

Подходящим способом интерпретируя A, B и **c** в этом законе, можно получить из него упомянутые высказывания и многие другие

При этом сам закон не зависит от выбранной интерпретации

Что такое логика

Законами логики задаются универсальные способы преобразования информации из одной формы в другую

Например:

«**У змей нет лап, и уж — змея**»

Упомянутый закон логики позволяет извлечь информацию, «скрытую» в этой фразе (форме), и представить её в другой форме:

«**У ужа нет лап**»

Следует иметь в виду, что при помощи законов логики **нельзя** получить новую информацию, а **можно** только преобразовать форму имеющейся информации

Иллюстрация того, насколько полезным может быть умение изменять форму информации:

«x **— наибольший простой делитель числá** $23\,082\,745\,709$»

и

«$x = 3221$» —

это две формы задания одного и того же числá x

Сто́ит ли уделять этому столько внимания?

Логика скрыта в основе многих явлений, процессов и задач, связанных с языком, рациональным познанием и умозаключениями

В лекциях — там, где это будет наиболее наглядно — иногда будут приводиться общие рассуждения и конкретные примеры по этому поводу

Но уже сейчас можно задуматься о простых для восприятия проблемах, с которыми непросто справиться без должных знаний
из области логики:

логических парадоксах

Несколько логических парадоксов

Парадокс лжеца

(*думаю, что все знают этот парадокс; а кто не знает — запоминайте*)

Это утверждение ложно

Ложно ли утверждение выше?

Несколько логических парадоксов

Парадокс Рассела

(*а этот парадокс должны знать все математики*)

Пусть Ω — множество всех множеств,
не содержащих себя в качестве элемента:

$$\Omega = \{\omega \mid \omega \notin \omega\}$$

Верно ли соотношение $\Omega \in \Omega$?

Несколько логических парадоксов

Парадокс утренней звезды

Венера видна ранним утром,
и поэтому её называют «утренней звездой»

Венера видна поздним вечером,
и поэтому её называют «вечерней звездой»

Означает ли это, что утренняя звезда
видна поздним вечером?

Несколько логических парадоксов

Парадокс лысого

(*он же «парадокс кучи»*)

Если у человека нет ни одного волоса, то он лыс

Если у лысого вырастет ещё один волосок, то он останется лысым

Значит, все люди лысые?

Несколько логических парадоксов

Парадокс пьяницы

Теорема. Все, кто здесь присутствует, пьют.

Доказательство.

Здесь присутствует человек, такой что если он пьёт, то все пьют. (*)

Я пью.

Значит, все пьют. ▼

Лемма. (*)

Доказательство.

Если все в этой комнате пьют,
то этот человек — любой из присутствующих (например, я).

Иначе один из присутствующих (**x**) — непьющий.

Пусть A — утверждение «**x** пьёт», и B — утверждение «все пьют».

Утверждение A ложно, а значит, утверждение «если A, то B» истинно. ▼

Где скрыта ошибка, и почему это ошибка?

Несколько логических парадоксов

Парадокс морской битвы

Флотоводец обратился к прорицателю с вопросом,
состоится ли завтра морская битва.
Прорицатель ответил: «Битва завтра состоится».

На следующий день случился шторм, и флот не смог выйти в море.
Разгневанный флотоводец потребовал от прорицателя
вернуть деньги, поскольку его прогноз оказался ложным.

Прорицатель ответил:
«Твои моряки вчера купили на рынке свежее молоко.
Сегодня это молоко уже не свежее,
но они не просят вернуть им деньги обратно.
Мой прогноз тоже был верным вчера,
и ты не вправе жаловаться на то, что он неверен сегодня».

Прав ли прорицатель?

Парадоксы неизбежны,
но их влияние можно ограничить

Чтобы это ограничение было возможно и неоспоримо, необходимо научиться

- анализировать форму и смысл высказываний, составляющих парадокс, и, более того
- делать это со всей (математической) строгостью

Для этого (*как и для многого другого*) и предназначена

МАТЕМАТИЧЕСКАЯ ЛОГИКА

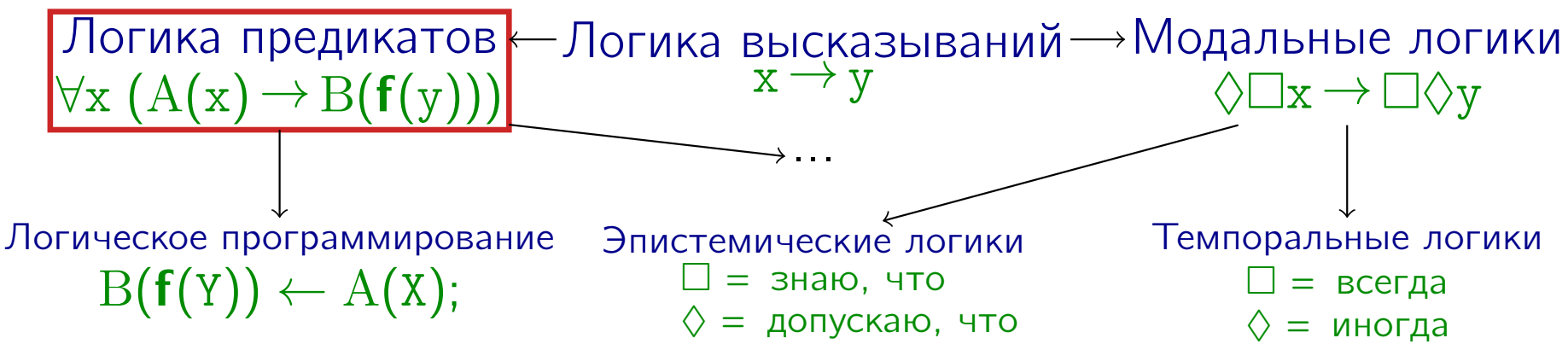
Чего ожидать в курсе

Использование логики, как правило, начинается с выбора языка записи интересующих высказываний

Таких языков существует великое множество, в зависимости от выбранного класса высказываний и целевых средств их анализа

В математической логике обычно используются искусственные формальные языки (языки формул)

В курсе встретятся следующие разделы логики с соответствующими формальными языками:



Чего ожидать в курсе

Основная задача логики, вокруг которой строится практически весь курс лекций — это задача проверки логического следования:

Правда ли, что высказывание A
обязательно следует из совокупности высказываний Γ?

$$\Gamma \models \mathrm{A}?$$

Эта задача возникает во многих областях разумной деятельности — например:

- Экспертные системы:
  - Γ — база знаний, A — запрос,
  - аппарат применения логических законов — ядро системы
- Средства автоматизации доказательства теорем:
  - Γ — аксиомы, леммы и т.п., A — формулировка теоремы
  - аппарат применения логических законов — ядро средства
- Логическое программирование:
  - Γ — подпрограмма, A — её вызов,
  - логические законы — это средство интерпретации программ

Чего ожидать в курсе

Отдельно будет показано, что
проверка логического следования — это практически то же самое, что и проверка общезначимости (тождественной истинности):

Правда ли, что высказывание A — абсолютная истина?

$$\models \mathrm{A}\textbf{?}$$

В курсе будут обсуждаться два известных логических метода проверки общезначимости формул, каждый из которых по-своему раскрывает некоторые разделы и особенности устройства логики:

- метод семантических таблиц,
  универсальный и идеологически простой
- метод резолюций,
  специальный для логики предикатов и эффективный,
  и лежащий в основе интерпретаторов логических программ

Чего ожидать в курсе

Метод резолюций, применяющийся для ответа на вопрос «$\Gamma \models A?$», лежит в основе логической парадигмы программирования:

- Совокупность высказываний Γ образует логическую программу
- Высказывание A представляет собой входные данные: запрос к программе
- Справедливость соотношения $\Gamma \models A$ отвечает успешности выполнения программы

Чтобы такой язык программирования был полезным, потребуется ограничить язык логических формул и способ применения метода резолюций так, чтобы обеспечить одновременно

- возможность пошаговой интерпретации формул,
- алгоритмическую полноту языка и
- удобство его использования

Математическая логика и логическое программирование

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

Блок 2

Логика высказываний:
синтаксис, семантика

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

ВМК МГУ, 2023/2024, осенний семестр

Логика высказываний: вступление

В языке логики высказываний содержатся средства записи

- «простых» высказываний, которые можно **интерпретировать** как истинные или ложные, и
- «булевых» причинно-следственных связей между этими высказываниями

Для примера посмотрим внимательно на такое предложение:

Если будете прогуливать лекции,

то ничего хорошего из этого не выйдет

Логика высказываний: вступление

В языке логики высказываний содержатся средства записи

- «простых» высказываний, которые можно **интерпретировать** как истинные или ложные, и
- «булевых» причинно-следственных связей между этими высказываниями

Для примера посмотрим внимательно на такое предложение:

Если будете прогуливать лекции,

то ничего хорошего из этого не выйдет

$$\underbrace{\text{Если будете прогуливать лекции, то ничего хорошего из этого не выйдет}}_{A}$$

Можно ли сказать, что это «простое высказывание»?

Логика высказываний: вступление

В языке логики высказываний содержатся средства записи

- «простых» высказываний, которые можно **интерпретировать** как истинные или ложные, и
- «булевых» причинно-следственных связей между этими высказываниями

Для примера посмотрим внимательно на такое предложение:

Если будете прогуливать лекции,

то ничего хорошего из этого не выйдет

$$\rightarrow$$

Здесь есть причинно-следственная связь . . .

Логика высказываний: вступление

В языке логики высказываний содержатся средства записи

- «простых» высказываний, которые можно **интерпретировать** как истинные или ложные, и
- «булевых» причинно-следственных связей между этими высказываниями

Для примера посмотрим внимательно на такое предложение:

Если будете прогуливать лекции,

то ничего хорошего из этого не выйдет

$$\mathtt{A} \rightarrow \quad \mathtt{B}$$

. . . между двумя простыми высказываниями

Логика высказываний: вступление

В языке логики высказываний содержатся средства записи

- «простых» высказываний, которые можно **интерпретировать** как истинные или ложные, и
- «булевых» причинно-следственных связей между этими высказываниями

Для примера посмотрим внимательно на такое предложение:

Если будете прогуливать лекции,

то ничего хорошего из этого не выйдет

$$A \rightarrow \neg B$$

А одно из высказываний можно сделать ещё проще

Логика высказываний: вступление

В языке логики высказываний содержатся средства записи

- «простых» высказываний, которые можно **интерпретировать** как истинные или ложные, и
- «булевых» причинно-следственных связей между этими высказываниями

Для примера посмотрим внимательно на такое предложение:

Если будете прогуливать лекции,

то ничего хорошего из этого не выйдет

$$A \to \neg B$$

Логика высказываний: вступление

После формализации высказывания получилось что-то
очень похожее на **формулу булевой алгебры**, но не совсем:

Какой смысл имеет построенное высказывание?

Булева алгебра: значение формулы — это булева функция:

A	B	$A \to \neg B$
0	0	1
0	1	1
1	0	1
1	1	0

Логика высказываний: **?**

Основные три составляющие формального языка,
которые дальше будут введены и для языка логики высказываний:

- алфавит: набор символов, используемых в языке
- синтаксис: правила, по которым из символов строятся высказывания языка (**формулы**)
- семантика: значение этих высказываний

Логика высказываний: алфавит

Алфавит логики высказываний состоит из следующих символов:

1. Пропозициональные переменные
   - Будем считать, что в алфавите содержится счётное число таких переменных
   - Var — множество всех пропозициональных переменных

2. Логические связки:

Конъюнкция	(логическое И):	$\&$
Дизъюнкция	(логическое ИЛИ):	$\vee$
Отрицание	(логическое НЕ):	$\neg$
Импликация	(логическое ЕСЛИ-ТО):	$\rightarrow$

3. Скобки:

$$(\quad)$$

Синтаксис и БНФ

Синтаксис:

- *σύνταξις* (древнегреческий) — строй, организация, конструкция[1]
- система языковых категорий, относящихся к соединениям слов и строению предложений[2]

Для задания синтаксиса в курсе будут использоваться формы Бэкуса-Наура (БНФ):

что-то ::= *запись1* | *запись2* | ... | *записьN*

Прочтение такой БНФ:

- *Запись1* — это *что-то*
- *Запись2* — это *что-то*

  ...
- *ЗаписьN* — это *что-то*
- Других способов записи *чего-то* нет

1 Дворецкий. Древнегреческо-русский словарь

2 Ожегов, Шведова. Толковый словарь русского языка

Логика высказываний: синтаксис

БНФ, определяющая синтаксис формул логики высказываний:

$$\varphi \ ::= \ \mathtt{x} \ \mid \ (\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi),$$

где φ — формула и $\mathtt{x} \in \mathrm{Var}$

Формула вида $\mathtt{x}$, $\mathtt{x} \in \mathrm{Var}$, называется атомарной (атомом),
а остальные формулы $((\varphi \,\&\, \psi)$, $(\varphi \vee \psi)$, $(\varphi \to \psi)$, $(\neg\varphi))$ — составными

$\varphi = \psi$ — так в курсе будет обозначаться
посимвольное (синтаксическое) совпадение формул φ и ψ

Приоритет связок по убыванию: $\neg$, затем $\&$, затем $\vee$, затем $\to$

Скобки в записи формул можно опускать согласно приоритету связок,
а также согласно ассоциативности связок $\&$ и $\vee$:

$$A \,\&\, \neg B \,\&\, C \to D \vee E = ((A \,\&\, ((\neg B) \,\&\, C)) \to (D \vee E))$$

Логика высказываний: семантика

Семантика:

- $\sigma\eta\mu\alpha\nu\tau\upsilon\kappa\acute{o}\varsigma$ (древнегреческий) — обозначающий, значимый[1]
- значение, смысл языковой единицы[2]

Основные логические значения, на которых основана семантика формул: $\mathbb{t}$ — истина; $\mathbb{f}$ — ложь

Значение формулы однозначно определяется значениями её атомов

Значение $\mathcal{I}(\mathtt{x})$ атома $\mathtt{x}$ задаётся интерпретацией $\mathcal{I} : \mathrm{Var} \to \{\mathbb{t}, \mathbb{f}\}$

Значение $\mathcal{I}(\varphi)$ составной формулы φ в интерпретации $\mathcal{I}$ задаётся так:

$$\begin{aligned}
\mathcal{I}(\varphi \,\&\, \psi) = \mathbb{t} \quad &\Leftrightarrow \quad \mathcal{I}(\varphi) = \mathbb{t} \text{ и } \mathcal{I}(\psi) = \mathbb{t} \\
\mathcal{I}(\varphi \vee \psi) = \mathbb{t} \quad &\Leftrightarrow \quad \mathcal{I}(\varphi) = \mathbb{t} \text{ или } \mathcal{I}(\psi) = \mathbb{t} \\
\mathcal{I}(\varphi \to \psi) = \mathbb{t} \quad &\Leftrightarrow \quad \mathcal{I}(\varphi) = \mathbb{f} \text{ или } \mathcal{I}(\psi) = \mathbb{t} \\
\mathcal{I}(\neg\varphi) = \mathbb{t} \quad &\Leftrightarrow \quad \mathcal{I}(\varphi) = \mathbb{f}
\end{aligned}$$

В логике принято использовать немного другие обозначения:

$$\begin{aligned}
\mathcal{I} \models \varphi \quad &\Leftrightarrow \quad \mathcal{I}(\varphi) = \mathbb{t} \quad (\text{формула } \varphi \text{ выполняется в } \mathcal{I}) \\
\mathcal{I} \not\models \varphi \quad &\Leftrightarrow \quad \mathcal{I}(\varphi) = \mathbb{f} \quad (\text{формула } \varphi \text{ не выполняется в } \mathcal{I})
\end{aligned}$$

1 Дворецкий. Древнегреческо-русский словарь

2 Ожегов, Шведова. Толковый словарь русского языка

Логика высказываний: семантика

Пример

$\mathrm{Var} = \{\mathtt{A}, \mathtt{B}, \ldots\}$ $\qquad \varphi = \mathtt{A} \to \neg\mathtt{B}$ $\qquad \mathcal{I}(\mathtt{A}) = \mathbb{f}$, $\mathcal{I}(\mathtt{B}) = \mathbb{t}$

Имеет место следующее:

$\mathcal{I} \not\models \neg\mathtt{B}$ $\quad$ (так как $\mathcal{I}(\mathtt{B}) = \mathbb{t}$)

$\mathcal{I} \models \mathtt{A} \to \neg\mathtt{B}$ $\quad$ (так как $\mathcal{I}(\mathtt{A}) = \mathbb{f}$)

Содержательное пояснение

Пусть $\mathtt{A}$ — высказывание «Я прогуливаю лекции»

$\mathtt{B}$ — высказывание «Из этого выйдет что-то хорошее»

Тогда $\mathcal{I}$ — мир, в котором я живу:

$\mathcal{I}(\mathtt{A}) = \mathbb{f}$: я прилежно хожу на лекции

$\mathcal{I}(\mathtt{B}) = \mathbb{t}$: из этого выйдет что-то хорошее

$\mathcal{I} \models \mathtt{A} \to \neg\mathtt{B}$: тот, кто сказал

«Если я прогуливаю лекции,

то из этого не выйдет ничего хорошего», **прав**

Логика высказываний: семантика

Пример

$$\mathrm{Var} = \{\mathtt{A}, \mathtt{B}, \ldots\} \qquad \varphi = \mathtt{A} \to \neg\mathtt{B} \qquad \mathcal{I}(\mathtt{A}) = \mathbb{t},\ \mathcal{I}(\mathtt{B}) = \mathbb{f}$$

Имеет место следующее:

$\mathcal{I} \models \neg\mathtt{B}$ (так как $\mathcal{I}(\mathtt{B}) = \mathbb{f}$)

$\mathcal{I} \models \mathtt{A} \to \neg\mathtt{B}$ (так как $\mathcal{I}(\mathtt{A}) = \mathbb{f}$)

Содержательное пояснение

Пусть $\mathtt{A}$ — высказывание «Я прогуливаю лекции»

$\mathtt{B}$ — высказывание «Из этого выйдет что-то хорошее»

Тогда $\mathcal{I}$ — мир, в котором я живу:

$\mathcal{I}(\mathtt{A}) = \mathbb{t}$: я прилежно хожу на лекции

$\mathcal{I}(\mathtt{B}) = \mathbb{f}$: из этого не выйдет ничего хорошего

$\mathcal{I} \models \mathtt{A} \to \neg\mathtt{B}$: тот, кто сказал

«Если я прогуливаю лекции,

то из этого не выйдет ничего хорошего», **прав**

Логика высказываний: семантика

Пример

$\mathrm{Var} = \{\mathtt{A}, \mathtt{B}, \ldots\}$ $\qquad \varphi = \mathtt{A} \to \neg\mathtt{B}$ $\qquad \mathcal{I}(\mathtt{A}) = \mathbb{t}$, $\mathcal{I}(\mathtt{B}) = \mathbb{t}$

Имеет место следующее:

$\mathcal{I} \not\models \neg\mathtt{B}$ $\qquad$ (так как $\mathcal{I}(\mathtt{B}) = \mathbb{t}$)

$\mathcal{I} \not\models \mathtt{A} \to \neg\mathtt{B}$ $\qquad$ (так как $\mathcal{I}(\mathtt{A}) = \mathbb{t}$ *и* $\mathcal{I}(\neg\mathtt{B}) = \mathbb{f}$)

Содержательное пояснение

Пусть $\mathtt{A}$ — высказывание «Я прогуливаю лекции»

$\mathtt{B}$ — высказывание «Из этого выйдет что-то хорошее»

Тогда $\mathcal{I}$ — мир, в котором я живу:

$\mathcal{I}(\mathtt{A}) = \mathbb{t}$: я прогуливаю лекции

$\mathcal{I}(\mathtt{B}) = \mathbb{t}$: из этого выйдет что-то хорошее

$\mathcal{I} \not\models \mathtt{A} \to \neg\mathtt{B}$: тот, кто сказал

«Если я прогуливаю лекции,

то из этого не выйдет ничего хорошего», **неправ**

Математическая логика и логическое программирование

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

Блок 3

Логика предикатов:
синтаксис, семантика

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Попробуем формализовать такое высказывание:

Если кто-то прогуливает лекции,
то экзамен он не сдаст

Вступление

Попробуем формализовать такое высказывание:

Если кто-то прогуливает лекции,
то экзамен он не сдаст

логика высказываний:

$$\mathrm{Shirk} \rightarrow \neg \mathrm{Pass}$$

Вступление

Попробуем формализовать такое высказывание:

Если кто-то прогуливает лекции,
то экзамен он не сдаст

$$\forall x\ (\mathrm{Shirk}(x) \rightarrow \neg \mathrm{Pass}(x))$$

Вступление

Попробуем формализовать такое высказывание:

Если кто-то прогуливает лекции,
то экзамен не сдаст его сосед

логика предикатов:

$$\forall x\ (\mathrm{Shirk}(x) \rightarrow \neg \mathrm{Pass}(\mathbf{neighbour}(x)))$$

Логика предикатов изучает законы
причинно-следственной зависимости между утверждениями,
основанными на **отношениях** между произвольными предметами

Формальный язык логики предикатов
ориентирован на описание таких отношений

Немного о названии «логика предикатов»

Предикат:

- preadicatum (латинский) — сказанное (сказуемое), объявленное[1]
- понятие, определяющее предмет суждения (субъект)[2]

Кто-то прогуливает лекцию

(*субъект*) (*предикат*) (*объект*)

Чуть более общо и развёрнуто, предикат — это

- свойство, атрибут предмета, явления, события, ...
- отношение между явлениями, предметами, событиями, ...

1 Дворецкий. Латинско-русский словарь

2 Ожегов, Шведова. Толковый словарь русского языка

Немного о названии «логика предикатов»

I

классическая

логика

предикатов

первого порядка

Немного о названии «логика предикатов»

II

логика
предикатов
первого порядка

Немного о названии «логика предикатов»

III

логика
предикатов

Немного о названии «логика предикатов»

IV

логика

первого порядка

Немного о названии «логика предикатов»

Будем использовать такое название:

логика
предикатов

Логика предикатов: алфавит

Базовые символы

Предметные константы

Ими обозначаются конкретные (именованные, фиксированные) предметы

Например: **я**, **2**, $\boldsymbol{\pi}$, **Солнце**, $\mathbf{c}_1$, ...

Const — множество всех констант

Предметные переменные

Ими обозначаются безымянные (нефиксированные) предметы

Они будут записываться привычно: $\mathtt{x}$, $\mathtt{y}'$, $\mathtt{z}_4$, ...

Var — множество всех переменных

Далее это множество полагается счётным и заданным однозначно

Логика предикатов: алфавит

Базовые символы

Функциональные символы

Ими обозначаются операции над предметами

Например: $+$, **сосед**, **lim**, ...

Каждому функциональному символу сопоставляется особое натуральное число — местность

$\mathbf{f}^{(k)}$ — запись функционального символа $\mathbf{f}$ с обозначением местности k

Func — множество всех функциональных символов с сопоставленными им местностями

Предикатные символы

Ими обозначаются отношения между предметами и свойства предметов

Например: $<$, **является_соседом**, **красный**, ...

При задании языка логики предикатов каждому предикатному символу сопоставляется особое натуральное число — местность

$\mathrm{P}^{(k)}$ — запись предикатного символа P с обозначением местности k

Pred — множество всех предикатных символов с сопоставленными им местностями

Логика предикатов: алфавит

Логические операции

Логические связки $\quad \& \quad \vee \quad \neg \quad \rightarrow$

Кванторы

Квантор всеобщности («для любого предмета»): $\forall$

Квантор существования («существует предмет»): $\exists$

Знаки препинания $\quad (\quad) \quad ,$

Сигнатурой алфавита логики предикатов называется тройка

$$\langle \text{Const}, \text{Func}, \text{Pred} \rangle$$

Устройство языка логики предикатов
однозначно определяется выбором сигнатуры:
все символы, не обозначенные в сигнатуре, определены однозначно

Логика предикатов: синтаксис

БНФ, определяющая синтаксис формул логики предикатов:

$$
\begin{array}{rcl}
t & ::= & \mathrm{x} \mid \mathbf{c} \mid \mathbf{f}^{(n)}(t_1, t_2, \ldots, t_n) \\
\varphi & ::= & \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \mid \\
 & & (\varphi \,\&\, \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi) \mid (\neg\varphi) \mid \\
 & & (\forall \mathrm{x}\, \varphi) \mid (\exists \mathrm{x}\, \varphi),
\end{array}
$$

где:

- φ — формула
- $t, t_1, t_2, \ldots, t_n$ — термы
- $\mathrm{x} \in \mathrm{Var}$
- $\mathbf{c} \in \mathrm{Const}$
- $\mathbf{f}^{(n)} \in \mathrm{Func}$
- $\mathrm{P}^{(k)} \in \mathrm{Pred}$

Логика предикатов: синтаксис / термы

$$t ::= \mathtt{x} \mid \mathbf{c} \mid \mathbf{f}^{(n)}(t_1, t_2, \ldots, t_n)$$

При помощи термов описываются предметы, получающиеся в результате применения заданных функций (операций) к заданным предметам

Например: ($\mathtt{z} \in \mathrm{Var}$; $\mathbf{1} \in \mathrm{Const}$; $+^{(2)}, \cdot^{(2)} \in \mathrm{Func}$)

1. Предмет, обозначенный переменной $\mathtt{z}$:
$$\mathtt{z}$$
2. Предмет, обозначенный константой $\mathbf{1}$:
$$\mathbf{1}$$
3. Предмет, получающийся применением операции $+$ к (1) и (2):
$$+(\mathbf{1}, \mathtt{z})$$
4. Предмет, получающийся применением операции $\cdot$ к (1) и (3):
$$\cdot(\mathtt{z}, +(\mathbf{1}, \mathtt{z}))$$
5. Более наглядная инфиксная форма записи терма (4):
$$\mathtt{z} \cdot (\mathbf{1} + \mathtt{z})$$

Логика предикатов: синтаксис / термы

$$t ::= \mathtt{x} \mid \mathbf{c} \mid \mathbf{f}^{(n)}(t_1, t_2, \ldots, t_n)$$

Term — множество всех термов

(*над заданными множествами* Var, Const, Func)

$\widetilde{\mathtt{x}}^{\mathtt{n}}$ — сокращённая запись последовательности $\langle\!\langle \mathtt{x}_1, \ldots, \mathtt{x}_\mathtt{n} \rangle\!\rangle$

Если t — терм, то:

Var_t — множество всех переменных, входящих в терм t

$t(\widetilde{\mathtt{x}}^{\mathtt{n}})$ — синоним записи t, если $\mathrm{Var}_t \subseteq \{\mathtt{x}_1, \ldots, \mathtt{x}_\mathtt{n}\}$

Терм t — основной, если $\mathrm{Var}_t = \emptyset$

Пример: если $\mathtt{x} \in \mathrm{Var}$, $\mathbf{1}, \mathbf{3} \in \mathrm{Const}$ и $+^{(2)}, \cdot^{(2)} \in \mathrm{Func}$, то терм

$$\mathbf{3} \cdot (\mathbf{1} + \mathbf{3})$$

является основным, а терм

$$\mathbf{3} \cdot (\mathbf{1} + \mathtt{x})$$

не является

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid$$
$$(\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

При помощи формул описываются отношения между предметами, строящиеся из «базовых» отношений при помощи логических операций
В некоторых случаях (*отношение местности 0*) формулой может описываться и высказывание, оцениваемое как истина или ложь

Например: $(\mathrm{P}^{(2)}, \mathrm{R}^{(1)} \in \mathsf{Pred};\ \mathbf{f}^{(2)} \in \mathsf{Func};\ \mathrm{x}, \mathrm{y} \in \mathsf{Var})$

1. Предмет y и предмет, получающийся из предметов x и y применением операции $\mathbf{f}$, входят в отношение P:
$$\mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y}))$$
2. Для любого предмета x верно (1):
$$(\forall \mathrm{x}\ \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y})))$$
3. Если верно (2), то предмет y обладает свойством R:
$$((\forall \mathrm{x}\ \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y}))) \to \mathrm{R}(\mathrm{y}))$$
4. Хотя бы для одного предмета y верно (3)
$$(\exists \mathrm{y}\ ((\forall \mathrm{x}\ \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y}))) \to \mathrm{R}(\mathrm{y})))$$

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid$$
$$(\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

Формула атомарна (является атомом), если имеет вид $\mathrm{P}(t_1, t_2, \ldots, t_k)$, где $\mathrm{P}^{(k)} \in \mathrm{Pred}$ и $t_1, t_2, \ldots, t_k \in \mathrm{Term}$

Остальные формулы называются составными

Form — множество всех формул
(*в алфавите с заданной сигнатурой*)

Приоритет логических операций (в порядке убывания):

$\forall$, $\exists$, $\neg$; затем $\&$; затем $\vee$; затем $\to$

Как работают приоритеты (пример)

Следующие формулы считаются синтаксически одинаковыми:

$$\forall \mathrm{x}\ \neg\mathrm{P}(\mathrm{x}) \,\&\, \exists \mathrm{y}\ \mathrm{R}(\mathrm{x},\mathrm{y}) \to \exists \mathrm{x}\ (\neg\mathrm{P}(\mathrm{x}) \vee \mathrm{P}(\mathrm{y}))$$
$$\forall \mathrm{x}\ (\neg\mathrm{P}(\mathrm{x})) \,\&\, (\exists \mathrm{y}\ \mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{x}\ ((\neg\mathrm{P}(\mathrm{x})) \vee \mathrm{P}(\mathrm{y}))$$
$$(\forall \mathrm{x}\ (\neg\mathrm{P}(\mathrm{x}))) \,\&\, (\exists \mathrm{y}\ \mathrm{R}(\mathrm{x},\mathrm{y})) \to (\exists \mathrm{x}\ ((\neg\mathrm{P}(\mathrm{x})) \vee \mathrm{P}(\mathrm{y})))$$
$$((\forall \mathrm{x}\ (\neg\mathrm{P}(\mathrm{x}))) \,\&\, (\exists \mathrm{y}\ \mathrm{R}(\mathrm{x},\mathrm{y}))) \to (\exists \mathrm{x}\ ((\neg\mathrm{P}(\mathrm{x})) \vee \mathrm{P}(\mathrm{y})))$$
$$(((\forall \mathrm{x}\ (\neg\mathrm{P}(\mathrm{x}))) \,\&\, (\exists \mathrm{y}\ \mathrm{R}(\mathrm{x},\mathrm{y}))) \to (\exists \mathrm{x}\ ((\neg\mathrm{P}(\mathrm{x})) \vee \mathrm{P}(\mathrm{y}))))$$

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid$$
$$(\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

Квантор связывает ту переменную, которая следует за ним

Область действия внешнего квантора
в формуле $\exists \mathrm{x}\ \varphi$ — это подформула φ

Вхождение переменной в область действия
связывающего её квантора — связанное вхождение

Вхождение переменной, не являющееся связанным, —
свободное вхождение

Переменная, имеющая свободное вхождение, —
свободная переменная формулы

Пример:

$$\exists \mathrm{y}(\forall \mathrm{x} \neg \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y})) \to \mathrm{R}(\mathrm{x}))$$

Логика предикатов: синтаксис / формулы

$$\varphi ::= \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \mid (\varphi \,\&\, \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi) \mid (\neg\varphi) \mid (\forall \mathrm{x}\ \varphi) \mid (\exists \mathrm{x}\ \varphi)$$

Квантор связывает ту переменную, которая следует за ним

Область действия внешнего квантора
в формуле $\exists \mathrm{x}\ \varphi$ — это подформула φ

Вхождение переменной в область действия
связывающего её квантора — связанное вхождение

Вхождение переменной, не являющееся связанным, —
свободное вхождение

Переменная, имеющая свободное вхождение, —
свободная переменная формулы

Пример:

$$\exists \mathrm{y}(\forall \mathrm{x} \neg \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y})) \to \mathrm{R}(\mathrm{x}))$$

Переменная y связана квантором $\exists$

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid$$
$$(\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

Квантор связывает ту переменную, которая следует за ним

Область действия внешнего квантора
в формуле $\exists \mathrm{x}\ \varphi$ — это подформула φ

Вхождение переменной в область действия
связывающего её квантора — связанное вхождение

Вхождение переменной, не являющееся связанным, —
свободное вхождение

Переменная, имеющая свободное вхождение, —
свободная переменная формулы

Пример:

$$\exists \mathrm{y}(\forall \mathrm{x} \neg \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y})) \to \mathrm{R}(\mathrm{x}))$$

Переменная x связана квантором $\forall$

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid$$
$$(\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

Квантор связывает ту переменную, которая следует за ним

Область действия внешнего квантора
в формуле $\exists \mathrm{x}\ \varphi$ — это подформула φ

Вхождение переменной в область действия
связывающего её квантора — связанное вхождение

Вхождение переменной, не являющееся связанным, —
свободное вхождение

Переменная, имеющая свободное вхождение, —
свободная переменная формулы

Пример:

$$\exists \mathrm{y}(\forall \mathrm{x} \neg \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y})) \to \mathrm{R}(\mathrm{x}))$$

Область действия квантора $\exists$

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid$$
$$(\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

Квантор связывает ту переменную, которая следует за ним

Область действия внешнего квантора
в формуле $\exists \mathrm{x}\ \varphi$ — это подформула φ

Вхождение переменной в область действия
связывающего её квантора — связанное вхождение

Вхождение переменной, не являющееся связанным, —
свободное вхождение

Переменная, имеющая свободное вхождение, —
свободная переменная формулы

Пример:

$$\exists \mathrm{y}(\forall \mathrm{x} \neg \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y})) \to \mathrm{R}(\mathrm{x}))$$

Область действия квантора $\forall$

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid$$
$$(\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

Квантор связывает ту переменную, которая следует за ним

Область действия внешнего квантора
в формуле $\exists \mathrm{x}\ \varphi$ — это подформула φ

Вхождение переменной в область действия
связывающего её квантора — связанное вхождение

Вхождение переменной, не являющееся связанным, —
свободное вхождение

Переменная, имеющая свободное вхождение, —
свободная переменная формулы

Пример:

$$\exists \mathrm{y}(\forall \mathrm{x} \neg \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y})) \to \mathrm{R}(\mathrm{x}))$$

Связанные вхождения переменной y

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid$$
$$(\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

Квантор связывает ту переменную, которая следует за ним

Область действия внешнего квантора
в формуле $\exists \mathrm{x}\ \varphi$ — это подформула φ

Вхождение переменной в область действия
связывающего её квантора — связанное вхождение

Вхождение переменной, не являющееся связанным, —
свободное вхождение

Переменная, имеющая свободное вхождение, —
свободная переменная формулы

Пример:

$$\exists \mathrm{y}(\forall \mathrm{x} \neg \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y})) \to \mathrm{R}(\mathrm{x}))$$

Связанное вхождение переменной x

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid$$
$$(\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

Квантор связывает ту переменную, которая следует за ним

Область действия внешнего квантора
в формуле $\exists \mathrm{x}\ \varphi$ — это подформула φ

Вхождение переменной в область действия
связывающего её квантора — связанное вхождение

Вхождение переменной, не являющееся связанным, —
свободное вхождение

Переменная, имеющая свободное вхождение, —
свободная переменная формулы

Пример:

$$\exists \mathrm{y}(\forall \mathrm{x} \neg \mathrm{P}(\mathrm{y}, \mathbf{f}(\mathrm{x}, \mathrm{y})) \to \mathrm{R}(\mathrm{x}))$$

Свободное вхождение переменной x

Логика предикатов: синтаксис / формулы

$$\varphi \ ::= \ \mathrm{P}^{(k)}(t_1, t_2, \ldots, t_k) \ \mid \ (\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi) \ \mid \ (\forall \mathrm{x}\ \varphi) \ \mid \ (\exists \mathrm{x}\ \varphi)$$

Var_φ — множество всех свободных переменных формулы φ

Если φ — формула, то:

- $\varphi(\widetilde{\mathrm{x}^{\mathrm{n}}})$ — синоним записи φ, если $\mathrm{Var}_\varphi \subseteq \{\mathrm{x}_1, \ldots, \mathrm{x}_\mathrm{n}\}$
- если $\mathrm{Var}_\varphi = \emptyset$, то φ — замкнутая формула, или предложение

CForm^1 — множество всех замкнутых формул
(*в алфавите с заданной сигнатурой*)

1 Closed Formulae

Логика предикатов: семантика

Как и в логике высказываний, смысл формуле логики предикатов придаёт интерпретация — «мир, в котором живёт формула»

Интерпретация состоит из

- предметов, населяющих мир
- операций над предметами
  - (это смысл **функциональных символов**)
- отношений, связывающих предметы
  - (это смысл **предикатных символов**)

Таким образом, в основе интерпретаций логики предикатов лежат алгебраические системы[1]

---

1 не следует пугаться этого термина;
это и есть совокупность «предметы + операции + отношения»

Логика предикатов: семантика

Интерпретация (*сигнатуры* $\langle \mathrm{Const}, \mathrm{Func}, \mathrm{Pred} \rangle$) — это система $\langle \mathrm{D}, \overline{\mathrm{Const}}, \overline{\mathrm{Func}}, \overline{\mathrm{Pred}} \rangle$, где:

- D — непустое множество предметов
  (область интерпретации; предметная область; универсум)
- $\overline{\mathrm{Const}} : \mathrm{Const} \to \mathrm{D}$ — оценка констант
- $\overline{\mathrm{Func}} : \mathrm{Func} \to \cup_{n \geq 1}(\mathrm{D}^n \to \mathrm{D})$ — оценка функциональных символов
- $\overline{\mathrm{Pred}} : \mathrm{Pred} \to \cup_{n \geq 1}(\mathrm{D}^n \to \{\mathbb{t}, \mathbb{f}\})$ — оценка предикатных символов

$\overline{\mathbf{c}} = \overline{\mathrm{Const}}(\mathbf{c})$ — предмет, сопоставленный константе $\mathbf{c}$

$\overline{\mathbf{f}} = \overline{\mathrm{Func}}(\mathbf{f}) : \mathrm{D}^n \to \mathrm{D}$ — функция, сопоставленная символу $\mathbf{f}^{(n)}$

$\overline{\mathrm{P}} = \overline{\mathrm{Pred}}(\mathrm{P}) : \mathrm{D}^n \to \{\mathbb{t}, \mathbb{f}\}$ — предикат, сопоставленный символу $\mathrm{P}^{(n)}$

Логика предикатов: семантика

Пример

Сигнатура: Const $= \{\mathbf{c_1}, \mathbf{c_2}\}$, Func $= \{\mathbf{f}^{(1)}\}$, Pred $= \{\mathrm{P}^{(1)}, \mathrm{R}^{(2)}\}$

Интерпретация:

предметная область: D $= \{0, 1, 2\}$

оценка констант: $\overline{\mathbf{c_1}} = 0$, $\overline{\mathbf{c_2}} = 1$

оценка функциональных и предикатных символов:

$\overline{\mathbf{f}}(\mathrm{x})$

x	$\overline{\mathbf{f}}(\mathrm{x})$
0	1
1	2
2	0

$\overline{\mathrm{P}}(\mathrm{x})$

x	$\overline{\mathrm{P}}(\mathrm{x})$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{\mathrm{R}}(\mathrm{x}, \mathrm{y})$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

Логика предикатов: семантика термов

Значение $t(\widetilde{\mathtt{x}}^{\mathtt{n}})[\widetilde{d}^n]$ терма $t(\widetilde{\mathtt{x}}^{\mathtt{n}})$ в интерпретации $\mathcal{I}$ на наборе предметов $d_1, \ldots, d_n$ из области интерпретации — это предмет, задаваемый так:

- для терма-переменной $\mathtt{x}_i$:
$$\mathtt{x}_i[\widetilde{d}^n] = d_i$$
- для терма-константы $\mathbf{c}$:
$$\mathbf{c}[\widetilde{d}^n] = \overline{\mathbf{c}}$$
- для остальных термов:
$$\mathbf{f}(t_1, \ldots, t_k)[\widetilde{d}^n] = \overline{\mathbf{f}}(t_1[\widetilde{d}^n], \ldots, t_k[\widetilde{d}^n])$$

Логика предикатов: семантика формул

Отношение выполнимости формулы $\varphi(\widetilde{\mathrm{x}^{\mathrm{n}}})$ в интерпретации $\mathcal{I}$ на наборе предметов $d_1, \ldots, d_n$ из области интерпретации ($\mathcal{I} \models \varphi(\widetilde{\mathrm{x}^{\mathrm{n}}})[\widetilde{d^n}]$) определяется так:

- атомарная формула:
  $\mathcal{I} \models \mathrm{P}(t_1, \ldots, t_k)[\widetilde{d^n}]$
  $\Leftrightarrow$
  $\overline{\mathrm{P}}(t_1[\widetilde{d^n}], \ldots, t_k[\widetilde{d^n}]) = \mathbb{t}$

- отрицание:
  $\mathcal{I} \models (\neg\varphi)[\widetilde{d^n}]$
  $\Leftrightarrow$
  $\mathcal{I} \not\models \varphi[\widetilde{d^n}]$

Логика предикатов: семантика формул

Отношение выполнимости формулы $\varphi(\widetilde{x}^n)$ в интерпретации $\mathcal{I}$ на наборе предметов $d_1, \ldots, d_n$ из области интерпретации ($\mathcal{I} \models \varphi(\widetilde{x}^n)[\widetilde{d}^n]$) определяется так:

- конъюнкция:
  $\mathcal{I} \models (\varphi \,\&\, \psi)[\widetilde{d}^n]$
  $\Leftrightarrow$
  $\mathcal{I} \models \varphi[\widetilde{d}^n]$ и $\mathcal{I} \models \psi[\widetilde{d}^n]$
- дизъюнкция:
  $\mathcal{I} \models (\varphi \vee \psi)[\widetilde{d}^n]$
  $\Leftrightarrow$
  $\mathcal{I} \models \varphi[\widetilde{d}^n]$ или $\mathcal{I} \models \psi[\widetilde{d}^n]$
- импликация:
  $\mathcal{I} \models (\varphi \to \psi)[\widetilde{d}^n]$
  $\Leftrightarrow$
  $\mathcal{I} \not\models \varphi[\widetilde{d}^n]$ или $\mathcal{I} \models \psi[\widetilde{d}^n]$

Логика предикатов: семантика формул

Отношение выполнимости формулы $\varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})$ в интерпретации $\mathcal{I}$ на наборе предметов $d_1, \ldots, d_n$ из области интерпретации ($\mathcal{I} \models \varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})[\widetilde{d}^n]$) определяется так:

- квантор всеобщности:

  $\mathcal{I} \models (\forall \mathrm{x}_0\ \varphi(\mathrm{x}_0, \widetilde{\mathrm{x}}^{\mathrm{n}}))[\widetilde{d}^n] \quad \Leftrightarrow$

  для любого предмета d_0 из области интерпретации верно $\mathcal{I} \models \varphi(\mathrm{x}_0, \widetilde{\mathrm{x}}^{\mathrm{n}})[d_0, \widetilde{d}^n]$

- квантор существования:

  $\mathcal{I} \models (\exists \mathrm{x}_0\ \varphi(\mathrm{x}_0, \widetilde{\mathrm{x}}^{\mathrm{n}}))[\widetilde{d}^n] \quad \Leftrightarrow$

  хотя бы для одного предмета d_0 из области интерпретации верно $\mathcal{I} \models \varphi(\mathrm{x}_0, \widetilde{\mathrm{x}}^{\mathrm{n}})[d_0, \widetilde{d}^n]$

$\varphi[\mathrm{x}_1/d_1, \ldots, \mathrm{x}_{\mathrm{n}}/d_n]$ — синоним записи $\varphi(\mathrm{x}_1, \ldots, \mathrm{x}_{\mathrm{n}})[d_1, \ldots, d_n]$

В записи «$\mathcal{I} \models \varphi[]$» отношения выполнимости предложения φ на пустом наборе предметов будем, как правило, опускать «пустые» квадратные скобки и писать просто $\mathcal{I} \models \varphi$

Семантика: примеры

Рассмотрим интерпретации такого вида:

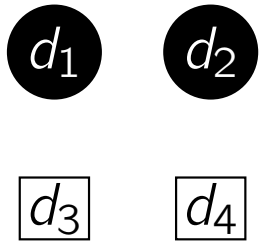
предметная область — квадраты и круги белого и чёрного цвета, расположенные на плоскости

сигнатура состоит из пяти предикатных символов, отвечающих следующим свойствам:

$\mathrm{C}(\mathrm{x})$: «x — круг»
$\mathrm{S}(\mathrm{x})$: «x — квадрат»
$\mathrm{B}(\mathrm{x})$: «x — чёрный предмет»
$\mathrm{W}(\mathrm{x})$: «x — белый предмет»
$\mathrm{U}(\mathrm{x}, \mathrm{y})$: «предмет x лежит под предметом y»

Семантика: примеры

Рассмотрим такую интерпретацию $\mathcal{I}$:



и такую формулу φ:

$$\forall \mathtt{x}\,(\mathrm{W}(\mathtt{x})\,\&\,\mathrm{S}(\mathtt{x}) \to \exists \mathtt{y}\,(\mathrm{B}(\mathtt{y})\,\&\,\mathrm{C}(\mathtt{y})\,\&\,\mathrm{U}(\mathtt{x},\mathtt{y})))$$

Формула содержательно прочитывается так:

Для каждого предмета $\mathtt{x}$: если он является белым и является квадратом, то существует предмет $\mathtt{y}$, такой что он является чёрным, и он является кругом, и предмет $\mathtt{x}$ лежит под предметом $\mathtt{y}$

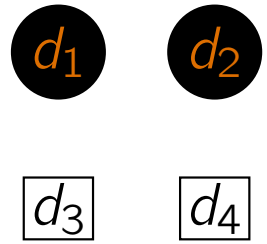
Проще говоря,

Каждый белый квадрат лежит под каким-то чёрным кругом

Чтобы строго убедиться, что это утверждение верно в $\mathcal{I}$, следует проверить соотношение $\mathcal{I} \models \varphi$

Семантика: примеры

Рассмотрим такую интерпретацию $\mathcal{I}$:



и такую формулу φ:

$$\forall \mathrm{x}\,(\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))$$

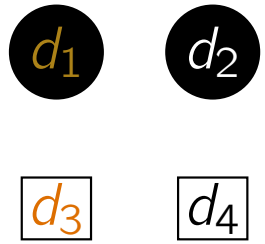
Переберём все предметы, подставляя их на место x

1. $\mathrm{x} \leftarrow d_1$:
   - $\mathcal{I} \not\models \mathrm{W}(\mathrm{x})[d_1]$
   - $\mathcal{I} \not\models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}))[d_1]$
   - $\mathcal{I} \models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_1]$

2. $\mathrm{x} \leftarrow d_2$:
   - $\mathcal{I} \not\models \mathrm{W}(\mathrm{x})[d_2]$
   - $\mathcal{I} \not\models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}))[d_2]$
   - $\mathcal{I} \models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_2]$

Семантика: примеры

Рассмотрим такую интерпретацию $\mathcal{I}$:



и такую формулу φ:

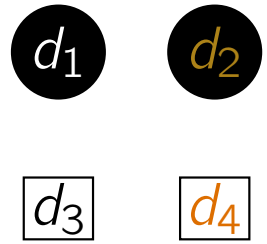
$$\forall \mathrm{x}\,(\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))$$

Переберём все предметы, подставляя их на место x

3. $\mathrm{x} \leftarrow d_3$:
   - $\mathcal{I} \models \mathrm{B}(\mathrm{y})[d_1]$
   - $\mathcal{I} \models \mathrm{C}(\mathrm{y})[d_1]$
   - $\mathcal{I} \models \mathrm{U}(\mathrm{x}, \mathrm{y})[\mathrm{y}/d_1, \mathrm{x}/d_3]$
   - $\mathcal{I} \models (\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y}))[\mathrm{y}/d_1, \mathrm{x}/d_3]$
   - $\mathcal{I} \models (\exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_3]$
   - $\mathcal{I} \models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_3]$

Семантика: примеры

Рассмотрим такую интерпретацию $\mathcal{I}$:



и такую формулу φ:

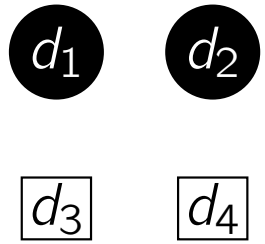
$$\forall \mathrm{x}\,(\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))$$

Переберём все предметы, подставляя их на место x

4. $\mathrm{x} \leftarrow d_4$:
   - $\mathcal{I} \models \mathrm{B}(\mathrm{y})[d_2]$
   - $\mathcal{I} \models \mathrm{C}(\mathrm{y})[d_2]$
   - $\mathcal{I} \models \mathrm{U}(\mathrm{x}, \mathrm{y})[\mathrm{y}/d_2, \mathrm{x}/d_4]$
   - $\mathcal{I} \models (\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y}))[\mathrm{y}/d_2, \mathrm{x}/d_4]$
   - $\mathcal{I} \models (\exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_4]$
   - $\mathcal{I} \models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_4]$

Семантика: примеры

Рассмотрим такую интерпретацию $\mathcal{I}$:



и такую формулу φ:

$$\forall \mathrm{x}\,(\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))$$

Итого:

$$\mathcal{I} \models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_1]$$

$$\mathcal{I} \models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_2]$$

$$\mathcal{I} \models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_3]$$

$$\mathcal{I} \models (\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))[d_4]$$

А значит,

$$\boxed{\mathcal{I} \models \forall \mathrm{x}\,(\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))}$$

Семантика: примеры

Рассмотрим такую интерпретацию $\mathcal{I}$:

сигнатура: $\mathrm{Const} = \emptyset$, $\mathrm{Func} = \{\mathbf{f}^{(1)}\}$, $\mathrm{Pred} = \{\mathrm{P}^{(1)}, \mathrm{R}^{(2)}\}$

предметная область: $\mathrm{D} = \{0, 1, 2\}$

оценки функциональных и предикатных символов:

$\bar{\mathbf{f}}(\mathrm{x})$

x	$\bar{\mathbf{f}}(\mathrm{x})$
0	1
1	2
2	0

$\overline{\mathrm{P}}(\mathrm{x})$

x	$\overline{\mathrm{P}}(\mathrm{x})$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{\mathrm{R}}(\mathrm{x}, \mathrm{y})$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

и такую формулу φ:

$$\forall \mathrm{x}\,(\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{R}(\mathrm{x}, \mathrm{y}) \,\&\, \neg \mathrm{P}(\mathbf{f}(\mathrm{y}))))$$

Проверим соотношение $\mathcal{I} \models \varphi$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models R(x,y)[y/0, x/2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models R(x,y)[y/0, x/2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\overline{\mathbf{f}}(x)$

x	$\overline{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\overline{\mathbf{f}}(x)$

x	$\overline{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

$\mathcal{I} \models P(\mathbf{f}(y))[1]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

$\mathcal{I} \models P(\mathbf{f}(y))[1]$

$\mathcal{I} \not\models (\neg P(\mathbf{f}(y)))[1]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

$\mathcal{I} \models P(\mathbf{f}(y))[1]$

$\mathcal{I} \not\models (\neg P(\mathbf{f}(y)))[1]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/1, x/2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\overline{\mathbf{f}}(x)$

x	$\overline{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/1, x/2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\overline{\mathbf{f}}(x)$

x	$\overline{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/1, x/2]$

$\mathcal{I} \models P(\mathbf{f}(y))[2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/1, x/2]$

$\mathcal{I} \models P(\mathbf{f}(y))[2]$

$\mathcal{I} \not\models (\neg P(\mathbf{f}(y)))[2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/1, x/2]$

$\mathcal{I} \models P(\mathbf{f}(y))[2]$

$\mathcal{I} \not\models (\neg P(\mathbf{f}(y)))[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/2, x/2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\overline{\mathbf{f}}(x)$

x	$\overline{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$$\mathcal{I} \models P(x)[2]$$

$$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$$

$$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/1, x/2]$$

$$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/2, x/2]$$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/0, x/2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/1, x/2]$

$\mathcal{I} \not\models (R(x,y)\,\&\,\neg P(\mathbf{f}(y)))[y/2, x/2]$

$\mathcal{I} \not\models (\exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))[2]$

Семантика: примеры

$$\forall \mathrm{x}\,(\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{R}(\mathrm{x},\mathrm{y}) \,\&\, \neg \mathrm{P}(\mathbf{f}(\mathrm{y}))))$$

$\bar{\mathbf{f}}(\mathrm{x})$

x	$\bar{\mathbf{f}}(\mathrm{x})$
0	1
1	2
2	0

$\overline{\mathrm{P}}(\mathrm{x})$

x	$\overline{\mathrm{P}}(\mathrm{x})$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{\mathrm{R}}(\mathrm{x},\mathrm{y})$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models \mathrm{P}(\mathrm{x})[2]$

$\mathcal{I} \not\models (\exists \mathrm{y}\,(\mathrm{R}(\mathrm{x},\mathrm{y}) \,\&\, \neg \mathrm{P}(\mathbf{f}(\mathrm{y}))))[2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$\mathcal{I} \models P(x)[2]$

$\mathcal{I} \not\models (\exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))[2]$

$\mathcal{I} \not\models (P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))[2]$

Семантика: примеры

$$\forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))$$

$\bar{\mathbf{f}}(x)$

x	$\bar{\mathbf{f}}(x)$
0	1
1	2
2	0

$\overline{P}(x)$

x	$\overline{P}(x)$
0	$\mathbb{t}$
1	$\mathbb{f}$
2	$\mathbb{t}$

$\overline{R}(x,y)$

x \ y	0	1	2
0	$\mathbb{t}$	$\mathbb{t}$	$\mathbb{f}$
1	$\mathbb{t}$	$\mathbb{f}$	$\mathbb{t}$
2	$\mathbb{f}$	$\mathbb{t}$	$\mathbb{t}$

$$\mathcal{I} \models P(x)[2]$$

$$\mathcal{I} \not\models (\exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))[2]$$

$$\mathcal{I} \not\models (P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))[2]$$

Значит,

$$\boxed{\mathcal{I} \not\models \forall x\,(P(x) \to \exists y\,(R(x,y)\,\&\,\neg P(\mathbf{f}(y))))}$$

Математическая логика и логическое программирование

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

Блок 4

Как формализовать предложение на языке логики предикатов (пример)

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

ВМК МГУ, 2023/2024, осенний семестр

Попробуем записать на языке логики предикатов
определение предела последовательности действительных чисел

Это определение формулируется для двух предметов:

1. Последовательность действительных чисел (`s`)
2. Действительное число (`x`),
   про которое говорится, что оно является пределом `s`

Вспомним, как определение записывается на естественном языке:

`s` — последовательность действительных чисел,
`x` — действительное число, и
для любого положительного действительного числа ε
существует натуральное число n, такое что
все элементы последовательности `s`, начиная с `n`-го,
отстоят от `x` не более чем на ε

Попробуем записать на языке логики предикатов
определение предела последовательности действительных чисел

Прежде чем начать записывать формулу, определимся с сигнатурой

Для примера выберем такую:

- $\mathbf{0} \in$ Const
- $\mathbf{ad}^{(2)} \in$ Func: $\quad \mathbf{ad}(\mathrm{x}, \mathrm{y}) = \text{«}|x - y|\text{»}$
- $\mathrm{R}^{(1)}, \mathrm{N}^{(1)}, \mathrm{S}^{(1)}, \mathrm{E}^{(3)}, <^{(2)}, \leq^{(2)} \in$ Pred:
  - $\mathrm{R}(\mathrm{x}) = \text{«}\mathtt{x}$ — действительное число»
  - $\mathrm{N}(\mathrm{x}) = \text{«}\mathtt{x}$ — натуральное число»
  - $\mathrm{S}(\mathrm{x}) = \text{«}\mathtt{x}$ — последовательность действительных чисел»
  - $\mathrm{E}(\mathrm{x}, \mathrm{n}, \mathrm{s}) = \text{«}\mathtt{x}$ — $\mathtt{n}$-й член последовательности $\mathtt{s}$»
  - $\mathrm{x} < \mathrm{y}$, $\mathrm{x} \leq \mathrm{y}$ — отношения неравенства чисел $\mathtt{x}$ и $\mathtt{y}$

Попробуем записать на языке логики предикатов
определение предела последовательности действительных чисел

$\mathtt{s}$ — последовательность действительных чисел,
$\mathtt{x}$ — действительное число, и
для любого положительного действительного числа ε
существует натуральное число n, такое что
все элементы последовательности $\mathtt{s}$, начиная с $\mathtt{n}$-го,
отстоят от $\mathtt{x}$ не более чем на ε

Здесь записаны три утверждения,
связанные (один раз явно, один раз неявно) союзом «и»

На языке логики предикатов это переписывается так («и» = «&»):

$$\mathrm{S}(\mathtt{s}) \,\&\, \mathrm{R}(\mathtt{x}) \,\&\, \varphi_1$$

С формулой φ_1 разберёмся отдельно

Попробуем записать на языке логики предикатов
определение предела последовательности действительных чисел

Для любого положительного действительного числа ε
существует натуральное число `n`, такое что
все элементы последовательности `s`, начиная с `n`-го,
отстоят от `x` не более чем на ε

«Для любого» = «$\forall$», и справа от $\forall$ обязательно стоит переменная, обозначающая **произвольный предмет**

Переформулируем предложение соответствующим образом:

Для любого **предмета** ε верно следующее:
если ε — положительное действительное число, то существует ...

$$\forall\varepsilon\ (\mathrm{R}(\varepsilon)\,\&(\mathbf{0}{<}\varepsilon){\to}\varphi_2)$$

Теперь отдельно разберёмся с формулой φ_2

Попробуем записать на языке логики предикатов
определение предела последовательности действительных чисел

Существует натуральное число $\mathtt{n}$, такое что
все элементы последовательности $\mathtt{s}$, начиная с $\mathtt{n}$-го,
отстоят от $\mathtt{x}$ не более чем на ε

«Существует» = «хотя бы один» = «$\exists$», и справа от $\exists$ обязательно стоит переменная, обозначающая **произвольный предмет**

Переформулируем предложение соответствующим образом:

Существует **предмет** $\mathtt{n}$, для которого верно следующее:
$\mathtt{n}$ — натуральное число, и кроме того, все элементы ...

$$\exists \mathtt{n}\ (\mathrm{N}(\mathtt{n}) \& \varphi_3)$$

Теперь отдельно разберёмся с формулой φ_3

Попробуем записать на языке логики предикатов
определение предела последовательности действительных чисел

Все элементы последовательности $\mathtt{s}$, начиная с $\mathtt{n}$-го, отстоят от $\mathtt{x}$ не более чем на ε

«Все» = «для любого» = «$\forall$»

Присвоим произвольному элементу, о котором говорится в предложении, имя ($\mathtt{y}$), и переформулируем предложение:

Для любого предмета $\mathtt{y}$ верно следующее:
если $\mathtt{y}$ совпадает с каким-либо элементом последовательности $\mathtt{s}$
с номером, не меньшим $\mathtt{n}$, то $\mathtt{y}$ отстоит от $\mathtt{x}$ не более чем на ε

$$\forall \mathtt{y}\ (\varphi_4 \to \mathbf{ad}(\mathtt{x}, \mathtt{y}) \leq \varepsilon)$$

Теперь отдельно разберёмся с формулой φ_4

Попробуем записать на языке логики предикатов
определение предела последовательности действительных чисел

$\mathtt{y}$ совпадает с каким-либо элементом последовательности $\mathtt{s}$ с номером, не меньшим $\mathtt{n}$

В формуле φ_4 «снаружи» располагается одна из операций

$$\&,\ \vee,\ \to,\ \neg,\ \forall,\ \exists$$

Чтобы понять, какая именно, переформулируем предложение так, чтобы «снаружи» располагалось одно из словосочетаний

«и», «или», «если-то», «не», «для любого», «существует»:

Существует предмет m, такой что m — натуральное число, и m не меньше n, и y — m-й элемент последовательности s

$$\exists \mathtt{m}\ (\mathrm{N}(\mathtt{m})\&(\mathtt{n} \le \mathtt{m})\&\mathrm{E}(\mathtt{y}, \mathtt{m}, \mathtt{s}))$$

Попробуем записать на языке логики предикатов
определение предела последовательности действительных чисел

Ответ:

$$
\begin{array}{l}
\mathrm{S}(\mathtt{s})\,\&\,\mathrm{R}(\mathtt{x})\,\& \\
\forall\varepsilon\ (\\
\quad \mathrm{R}(\varepsilon)\,\&(\mathbf{0}{<}\varepsilon)\to \\
\quad \exists\mathtt{n}\ (\\
\qquad \mathrm{N}(\mathtt{n})\,\& \\
\qquad \forall\mathtt{y}\ (\\
\qquad\quad \exists\mathtt{m}\ (\\
\qquad\qquad \mathrm{N}(\mathtt{m})\,\&(\mathtt{n}\le\mathtt{m})\,\&\,\mathrm{E}(\mathtt{y},\mathtt{m},\mathtt{s}) \\
\qquad\quad)\to \\
\qquad\quad \mathbf{ad}(\mathtt{x},\mathtt{y})\le\varepsilon \\
\qquad) \\
\quad) \\
)
\end{array}
$$

Математическая логика и логическое программирование

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

Блок 5

Логика высказываний (ЛВ):
выполнимые и общезначимые формулы

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

ВМК МГУ, 2023/2024, осенний семестр

Основные свойства формул,
которые обычно исследуются в логике высказываний (ЛВ):

Формула φ выполнима ($||\!= \varphi$[1]), если
существует интерпретация $\mathcal{I}$, такая что $\mathcal{I} \models \varphi$

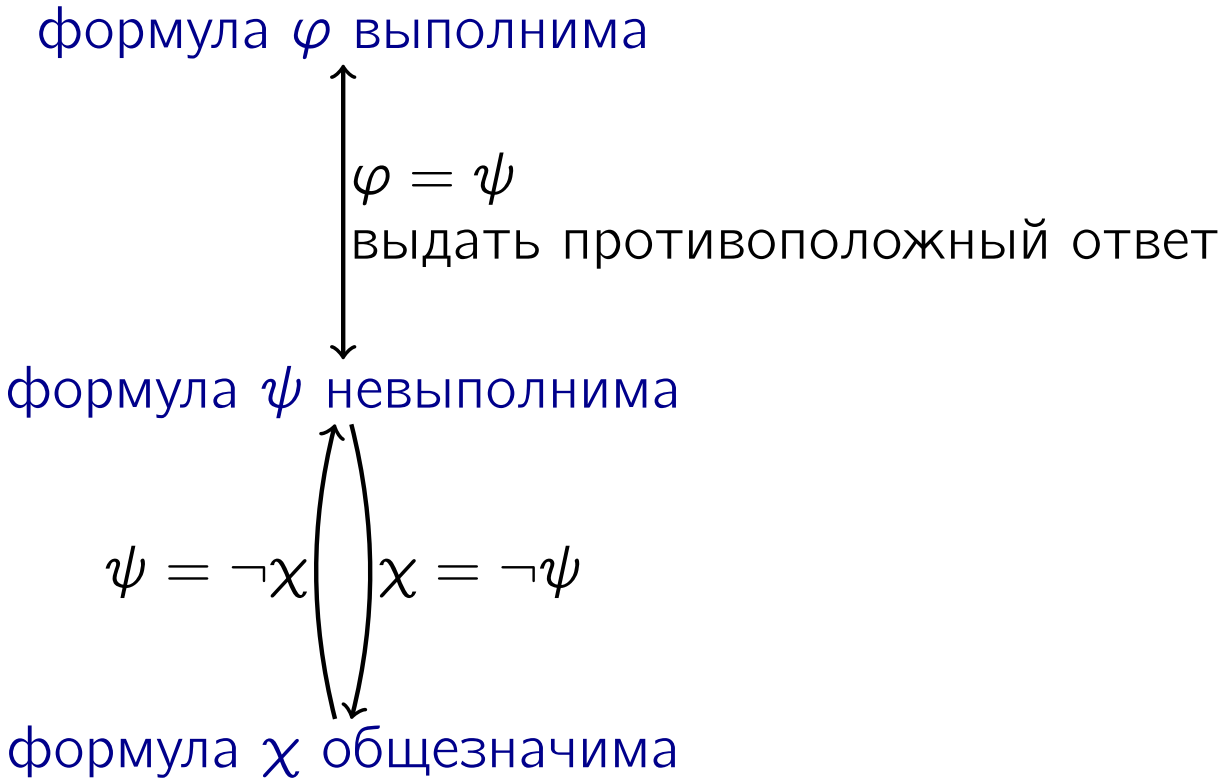
Формула φ общезначима ($\models \varphi$), если
для любой интерпретации $\mathcal{I}$ верно $\mathcal{I} \models \varphi$

Формула φ невыполнима ($|\!\not= \varphi$[1]), если она не является выполнимой,
и необщезначима ($\not\models \varphi$), если не является общезначимой

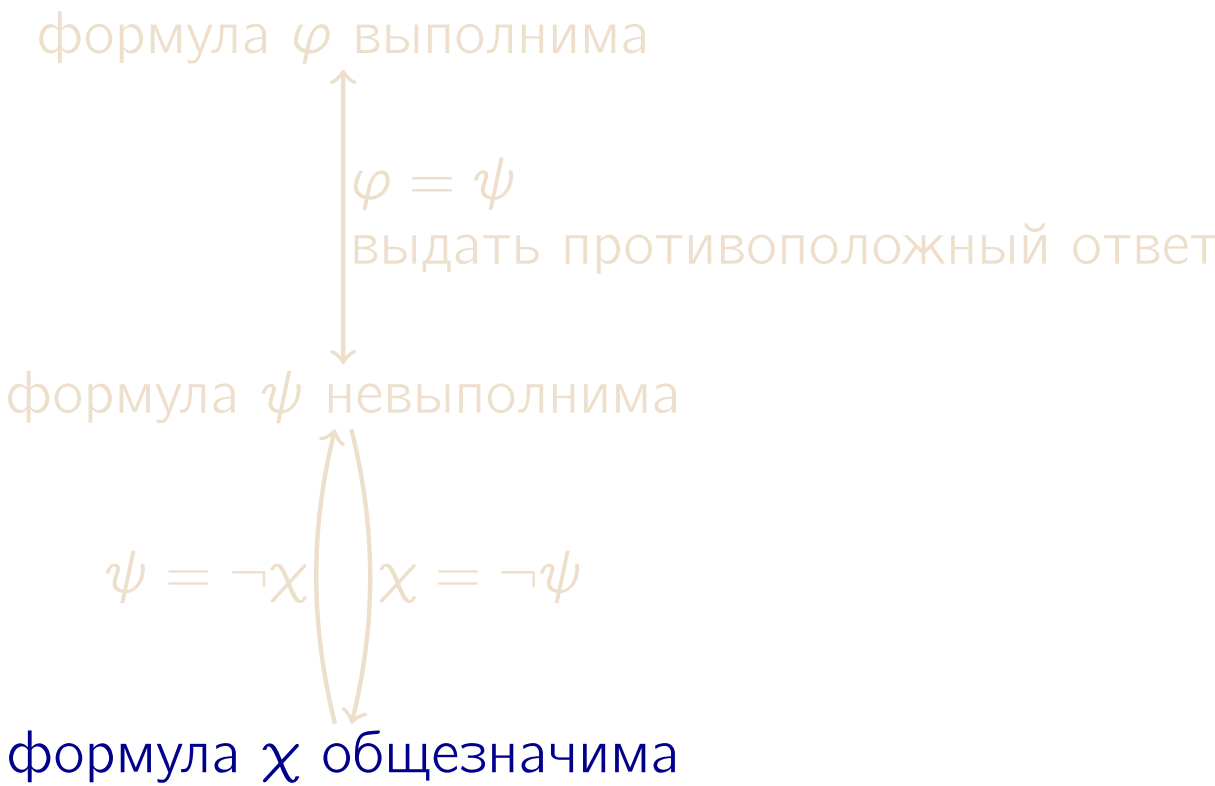
Логика высказываний		*Булева алгебра*
φ выполнима	$\Leftrightarrow$	φ выполнима
φ невыполнима	$\Leftrightarrow$	φ — тождественный ноль
φ общезначима	$\Leftrightarrow$	φ — тождественная единица

1 Это необщеупотребимое обозначение, которое придумал я
для использования в слайдах по этому курсу **и только здесь**.
Не используйте это обозначение, вас не поймут

Проверка каждого из этих свойств
(выполнимость, невыполнимость, общезначимость)
может быть легко **сведена** к проверке любого другого из них:



Проверка каждого из этих свойств
(выполнимость, невыполнимость, общезначимость)
может быть легко **сведена** к проверке любого другого из них:



Поэтому в логике нередко рассматривается только одно из этих свойств,
и обычно это свойство **общезначимости**

Математическая логика и логическое программирование

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

Блок 6

Логика предикатов (ЛП):
выполнимые и общезначимые формулы,
модели формул,
логическое следствие,
проблема общезначимости формул (постановка)

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Продолжаем обсуждение логики предикатов (ЛП)

Вспомним на примере, что есть что

Сигнатура:

$$\langle \{\mathbf{c}\}, \{\mathbf{f}^{(1)}\}, \{\mathrm{P}^{(1)}\} \rangle$$

Формула:

$$\varphi = \mathrm{P}(\mathbf{c}) \to \forall \mathrm{x}\, \mathrm{P}(\mathbf{f}(\mathrm{x}))$$

Интерпретация $\mathcal{I}$:

предметная область: $\{d_1, d_2\}$

$\bar{\mathbf{c}} = d_1 \qquad \bar{\mathbf{f}}(d_1) = \bar{\mathbf{f}}(d_2) = d_1 \qquad \overline{\mathrm{P}}(d_1) = \mathbb{t}, \quad \overline{\mathrm{P}}(d_2) = \mathbb{f}$

Отношение выполнимости:

$$\mathcal{I} \models \varphi$$

ЛП: выполнимые и общезначимые формулы

Формула $\varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})$ выполнима в интерпретации $\mathcal{I}$ ($\mathcal{I} \mid\!\models \varphi$[1]),
если **существует** набор предметов $\widetilde{d}^{n}$ из области интерпретации $\mathcal{I}$,
такой что $\mathcal{I} \models \varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})[\widetilde{d}^{n}]$

Формула $\varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})$ истинна в интерпретации $\mathcal{I}$ ($\mathcal{I} \models \varphi$),
если **для любого** набора предметов $\widetilde{d}^{n}$ из области интерпретации $\mathcal{I}$
верно $\mathcal{I} \models \varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})[\widetilde{d}^{n}]$

Формула φ выполнима ($\mid\!\models \varphi$[1]),
если существует интерпретация, в которой она выполнима

Формула φ общезначима
(тождественно истинна; является тавтологией; $\models \varphi$),
если она истинна в любой интерпретации

Про невыполнимую формулу также часто говорят,
что она тождественно ложна

---

1 Как и раньше, это необщеупотребимое обозначение

ЛП: выполнимые и общезначимые формулы

Пример

$$\begin{aligned} \varphi\colon &\quad \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \to \exists \mathrm{x}\, \mathrm{P}(\mathrm{x}) \\ \psi\colon &\quad \exists \mathrm{x}\, \mathrm{P}(\mathrm{x}) \to \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \\ \chi\colon &\quad \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \,\&\, \forall \mathrm{x}\, \neg \mathrm{P}(\mathrm{x}) \end{aligned}$$

Интерпретация $\mathcal{I}_1$: $\mathsf{D} = \{d\}$, $\overline{\mathrm{P}}(d) = \mathbb{t}$

$$\mathcal{I}_1 \models \varphi \qquad \mathcal{I}_1 \models \psi \qquad \mathcal{I}_1 \not\models \chi$$

Интерпретация $\mathcal{I}_2$: $\mathsf{D} = \{d_1, d_2\}$, $\overline{\mathrm{P}}(d_1) = \mathbb{t}$, $\overline{\mathrm{P}}(d_2) = \mathbb{f}$

$$\mathcal{I}_2 \models \varphi \qquad \mathcal{I}_2 \not\models \psi \qquad \mathcal{I}_2 \not\models \chi$$

Только что было показано, что

1. формулы φ, ψ выполнимы
2. формулы ψ, χ необщезначимы

А как доказать общезначимость φ и невыполнимость χ?

ЛП: выполнимые и общезначимые формулы

Выполнимые необщезначимые формулы — это логические формы, в которых записана некоторая «нетривиальная» («полезная») информация

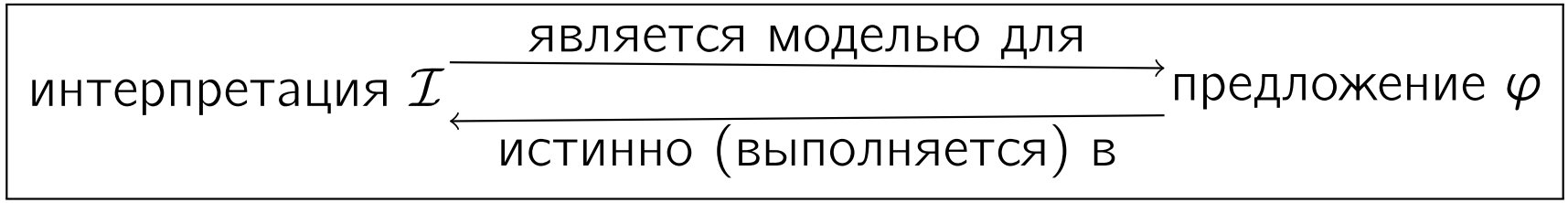
Общезначимые формулы — это (*казалось бы*) банальности, тавтологии, формы, не содержащие в себе никакой «полезной» информации

При этом общезначимые формулы в логике считаются чрезвычайно важными, и в курсе будут обсуждаться в основном такие формулы

Почему?

ЛП: модели

Интерпретация $\mathcal{I}$ называется моделью для **предложения** φ, если $\mathcal{I} \models \varphi$



Интерпретация $\mathcal{I}$ называется моделью для **множества предложений** Γ ($\mathcal{I} \models \Gamma$), если она является моделью для каждого предложения из Γ, и в этом случае говорят, что Γ выполняется в $\mathcal{I}$

Наряду с «модель для формулы/множества» будем также говорить «модель формулы/множества» (без «для»)

Относительно каждой интерпретации $\mathcal{I}$ все предложения делятся на

- выполнимые в $\mathcal{I}$ («верные») и
- невыполнимые в $\mathcal{I}$ («неверные»)

Относительно каждого предложения φ все интерпретации делятся на

- модели для φ (адекватно подходящие под устройство φ) и
- не являющиеся моделями для φ (неподходящие)

ЛП: модели

Пример

Снова рассмотрим интерпретации с квадратами и кругами белого и чёрного цвета на плоскости:

$\mathrm{C}(\mathrm{x})$: «x — круг» $\mathrm{B}(\mathrm{x})$: «x — чёрный предмет»

$\mathrm{S}(\mathrm{x})$: «x — квадрат» $\mathrm{W}(\mathrm{x})$: «x — белый предмет»

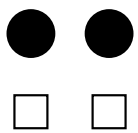
$\mathrm{U}(\mathrm{x}, \mathrm{y})$: «предмет x лежит под предметом y»

Рассмотрим такую формулу φ:

$$\forall \mathrm{x}\,(\mathrm{W}(\mathrm{x}) \,\&\, \mathrm{S}(\mathrm{x}) \to \exists \mathrm{y}\,(\mathrm{B}(\mathrm{y}) \,\&\, \mathrm{C}(\mathrm{y}) \,\&\, \mathrm{U}(\mathrm{x}, \mathrm{y})))$$

«**любой белый квадрат лежит под каким-то чёрным кругом**»

и такие интерпретации:

$\mathcal{I}_1$:  $\mathcal{I}_2$: 

Тогда $\mathcal{I}_1$ **является моделью** для φ, а $\mathcal{I}_2$ **не является**:

- $\mathcal{I}_1 \models \varphi$: оба белых квадрата лежат под чёрными кругами
- $\mathcal{I}_2 \not\models \varphi$: левый белый квадрат **не** лежит под чёрным кругом

ЛП: логическое следствие

Предложение φ называется логическим следствием множества предложений Γ ($\Gamma \models \varphi$), если любая модель Γ является моделью φ

Другими словами — если для любой интерпретации $\mathcal{I}$ верно

$$\mathcal{I} \models \Gamma \qquad \Rightarrow \qquad \mathcal{I} \models \varphi$$

Содержательно — если независимо от смысла символов сигнатуры из справедливости всех утверждений, записанных в Γ, **обязательно** следует справедливость утверждения φ

Отношение $\models$, используемое в таком смысле, будем называть отношением логического следования

Наряду с «$\{\psi_1, \ldots, \psi_n\} \models \varphi$» будем также писать «$\psi_1, \ldots, \psi_n \models \varphi$»

ЛП: логическое следствие

Небольшое пояснение:

- $\forall x\, P(x) \models P(\mathbf{c})$:
  если все предметы обладают свойством P, то **обязательно** предмет, обозначенный символом **c**, обладает свойством P
- $P(\mathbf{c}) \not\models \forall x\, P(x)$:
  если предмет, обозначенный символом **c**, обладает свойством P», то из этого **в общем случае не следует**, что все предеметы обладают свойством P

Одна из главных задач (и характерное проявление) интеллектуальной деятельности — это **извлечение логических следствий** из имеющихся баз знаний

Эта задача возникает в огромном числе областей «разумной деятельности»: экспертные системы, (автоматическое и ручное) доказательство теорем, формальный анализ программ, . . . , . . .

ЛП: логическое следствие (пример)

Покажем, что представляют собой логические следствия,
на простом показательном примере

Известно, что:

- Даша любит Сашу,
- а Саша любит пиво,
- а Паша любит пиво и всех тех, кто любит то же, что и он

Любит ли кто-нибудь Дашу?

Попробуем записать эту задачу на языке логики предикатов

Начнём с сигнатуры алфавита: в неё войдут

- константы **Даша**, **Саша**, **Паша**, **пиво** и
- предикатный символ $\mathrm{L}^{(2)}$: $\mathrm{L}(\mathrm{x}, \mathrm{y}) =$ «икс любит игрека»

ЛП: логическое следствие (пример)

Условия задачи переписываются так:

- Даша любит Сашу
$$\varphi_1 = \mathrm{L}(\textbf{Даша}, \textbf{Саша})$$
- Саша любит пиво
$$\varphi_2 = \mathrm{L}(\textbf{Саша}, \textbf{пиво})$$
- Паша любит пиво и всех тех, кто любит то же, что и он:
$$\begin{aligned} \varphi_3 &= \mathrm{L}(\textbf{Паша}, \textbf{пиво}) \\ \varphi_4 &= \forall \mathrm{x}\, (\exists \mathrm{y}\, (\mathrm{L}(\textbf{Паша}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y})) \to \mathrm{L}(\textbf{Паша}, \mathrm{x})) \end{aligned}$$
- Любит ли кто-нибудь дашу?

  Правда ли, что из знаний $\varphi_1, \ldots, \varphi_4$ необходимо следует знание
$$\varphi_0 = \exists x\, L(\mathrm{x}, \textbf{Даша})\ ?$$

В конечном итоге задача переписывается так:

проверить соотношение $\varphi_1, \varphi_2, \varphi_3, \varphi_4 \models \varphi_0$

Теорема о логическом следствии

Для любого предложения φ и любого конечного множества предложений $\Gamma = \{\psi_1, \psi_2, \ldots, \psi_n\}$ справедлива равносильность

$$\Gamma \models \varphi \quad \Leftrightarrow \quad \models \psi_1 \,\&\, \psi_2 \,\&\, \ldots \,\&\, \psi_n \to \varphi$$

Доказательство. ($\Rightarrow$) Предположим, что $\Gamma \models \varphi$

Рассмотрим **произвольную** интерпретацию $\mathcal{I}$

Если $\mathcal{I} \not\models \Gamma$, то $\mathcal{I} \not\models \psi_1 \,\&\, \ldots \,\&\, \psi_n$, а значит, $\mathcal{I} \models \psi_1 \,\&\, \ldots \,\&\, \psi_n \to \varphi$

Пусть теперь $\mathcal{I} \models \Gamma$

Так как $\Gamma \models \varphi$, верно и $\mathcal{I} \models \varphi$ —
а значит, снова верно $\mathcal{I} \models \psi_1 \,\&\, \ldots \,\&\, \psi_n \to \varphi$

Итог: для **любой** интерпретации $\mathcal{I}$ верно

$$\mathcal{I} \models \psi_1 \,\&\, \ldots \,\&\, \psi_n \to \varphi$$

По определению общезначимости, это означает $\models \psi_1 \,\&\, \ldots \,\&\, \psi_n \to \varphi$

Теорема о логическом следствии

Для любого предложения φ и любого конечного множества предложений $\Gamma = \{\psi_1, \psi_2, \ldots, \psi_n\}$ справедлива равносильность

$$\Gamma \models \varphi \qquad \Leftrightarrow \qquad \models \psi_1 \,\&\, \psi_2 \,\&\, \ldots \,\&\, \psi_n \to \varphi$$

Доказательство. ($\Leftarrow$) Предположим, что $\models \psi_1 \,\&\, \ldots \,\&\, \psi_n \to \varphi$

Рассмотрим **произвольную** модель $\mathcal{I}$ для множества Γ:

$$\mathcal{I} \models \psi_1, \qquad \ldots, \qquad \mathcal{I} \models \psi_n$$

Тогда $\mathcal{I} \models \psi_1 \,\&\, \ldots \,\&\, \psi_n$

Так как формула $\psi_1 \,\&\, \ldots \,\&\, \psi_n \to \varphi$ общезначима, верно

$$\mathcal{I} \models \psi_1 \,\&\, \ldots \,\&\, \psi_n \to \varphi$$

Согласно семантике $\to$, верно $\mathcal{I} \models \varphi$

Таким образом, **произвольная** модель $\mathcal{I}$ множества Γ является и моделью формулы φ, то есть $\Gamma \models \varphi$ ▼

ЛП: проблема общезначимости формул

Чтобы уметь извлекать логические следствия и в целом анализировать достоверность утверждений, необходимо понимать **законы**, связывающие достоверность различных утверждений

Общезначимые формулы представляют собой
один из способов записи таких законов — например, закон вида
«**если верны утверждения** $\psi_1, \ldots, \psi_n$**, то верно и** φ»
записывается в виде общезначимой формулы

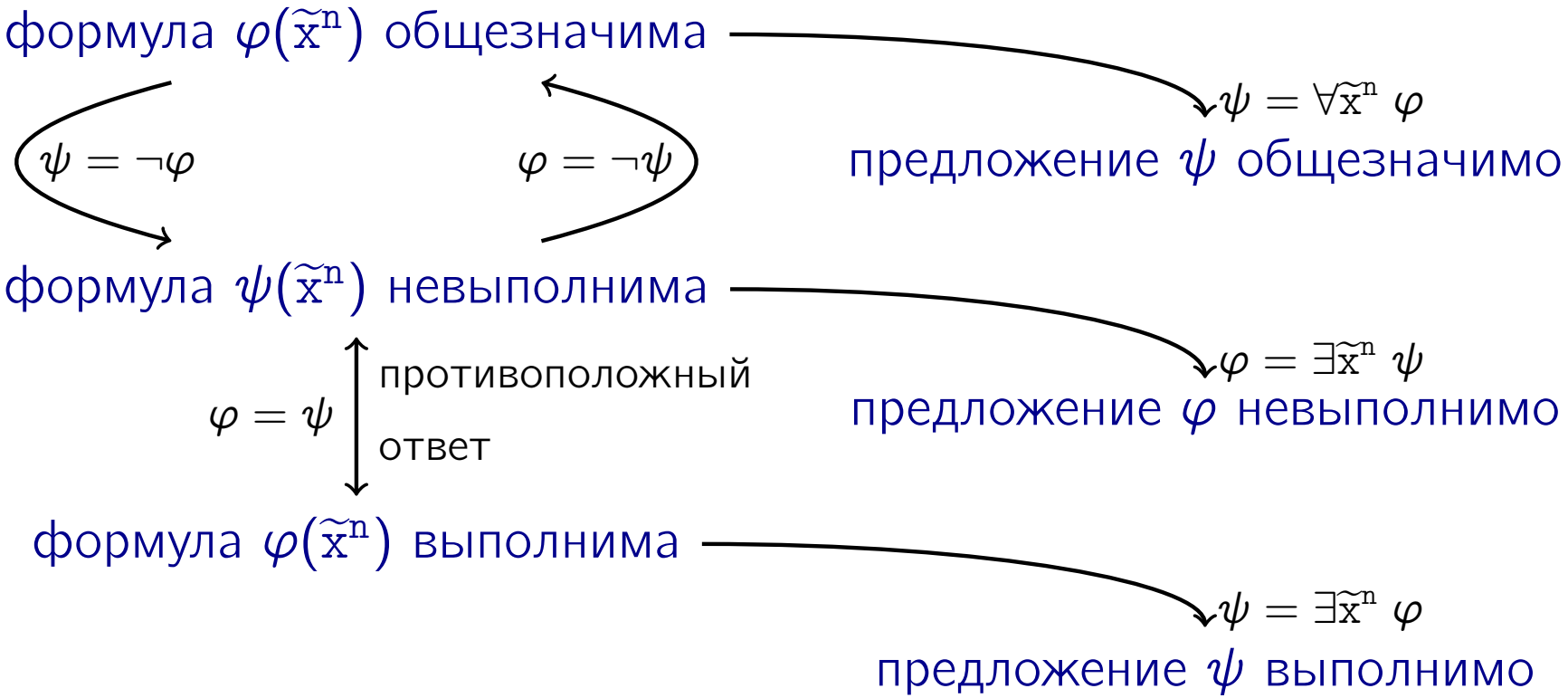
$$\psi_1 \,\&\, \ldots \,\&\, \psi_n \to \varphi$$

В связи с этим оказывается важна проблема общезначимости формул:

для заданной формулы φ
проверить её общезначимость:
$\models \varphi$?

ЛП: проблема общезначимости формул

Несколько слов о взаимосвязи свойств общезначимости, выполнимости и невыполнимости формул логики предикатов:



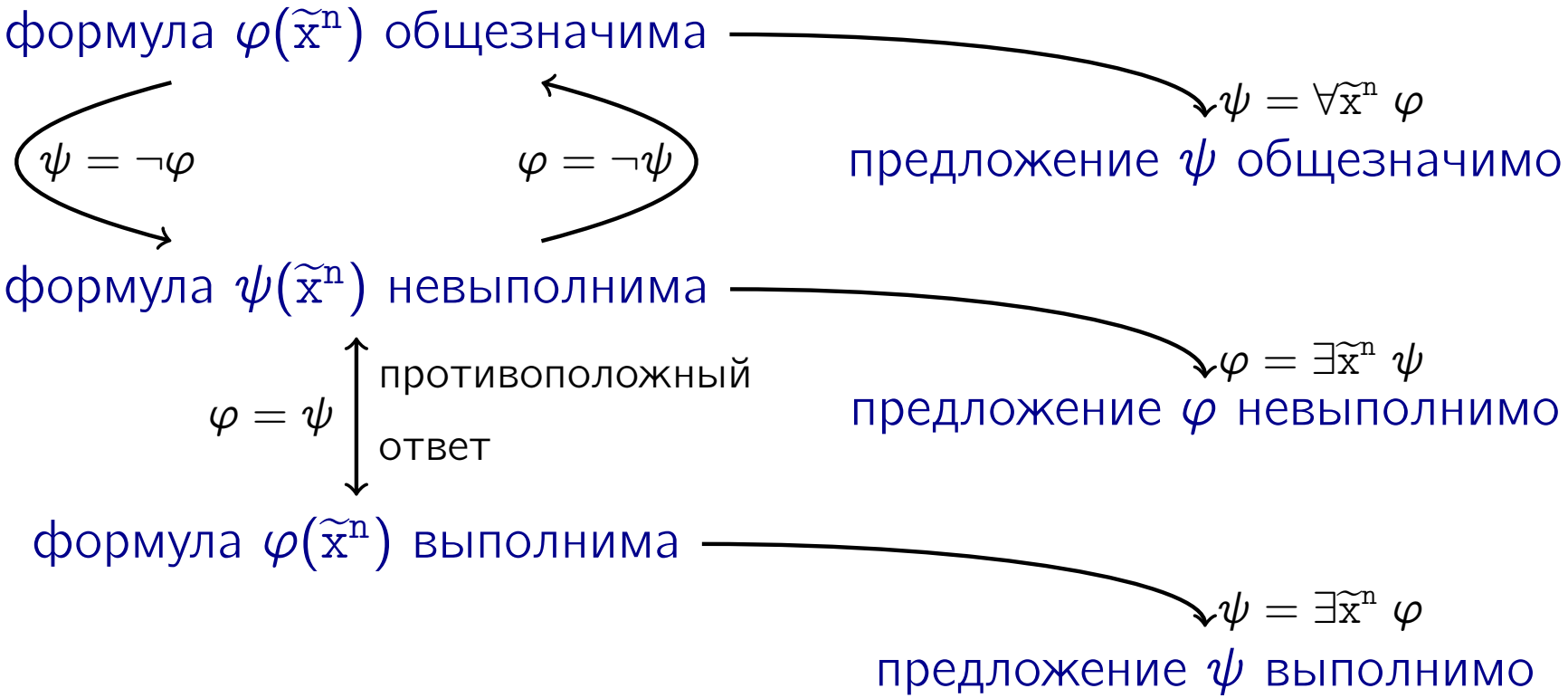
$\forall\widetilde{x}^n$ — сокращение для $\forall x_1 \ldots \forall x_n$

$\exists\widetilde{x}^n$ — сокращение для $\exists x_1 \ldots \exists x_n$

ЛП: проблема общезначимости формул

Несколько слов о взаимосвязи свойств общезначимости, выполнимости и невыполнимости формул логики предикатов:

Утверждение



Доказательство. Напрямую следует из определений выполнимости, невыполнимости и общезначимости ▼

Математическая логика и логическое программирование

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

Блок 7

Логика предикатов:
можно ли проверить
общезначимость формулы «в лоб»?

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

ВМК МГУ, 2023/2024, осенний семестр

Для логики высказываний существует очень простой способ проверки общезначимости формул «в лоб»:

- Вспомнить, что общезначимость формулы в этой логике — это то же самое, что равенство константе 1 в булевой алгебре
- **Перебрать** все наборы значений булевых переменных формулы и для каждого проверить, принимает ли реализуемая функция значение 0
- Если найден хотя бы один такой набор, то формула необщезначима; иначе формула общезначима

А можно ли адаптировать этот метод проверки общезначимости к логике предикатов?

«Перебрать все наборы значений булевых переменных формулы» — в ЛВ это «Перебрать все интерпретации формулы»

«Реализуемая функция принимает значение 0» — в ЛВ это «Формула не выполняется в интерпретации»

Значит, прямое обобщение «лобового» способа проверки общезначимости формулы φ на логику предикатов выглядит так:

- Перебрать всевозможные интерпретации
- Для каждой (очередной) интерпретации $\mathcal{I}$ проверить соотношение $\mathcal{I} \models \varphi$
- Если обнаружена интерпретация $\mathcal{I}$, такая что $\mathcal{I} \not\models \varphi$, то формула необщезначима
- Если все интерпретации перебраны и для каждой показано соотношение $\mathcal{I} \models \varphi$, то формула общезначима

Но всё не так просто:

- Как задать интерпретацию с бесконечной предметной областью?
- Как проверить истинность формулы в такой интерпретации?
- Как перебрать «ужасно»-бесконечное число интерпретаций?

Если бы можно было ограничиться только **конечными** интерпретациями, то проблем бы было намного меньше, но ...

Утверждение. **Существует необщезначимое предложение, истинное в любой интерпретации с конечной предметной областью**

Доказательство. Вот пример такого предложения φ:

$$\forall \mathrm{x}\ \neg \mathrm{R}(\mathrm{x}, \mathrm{x}) \,\&\, \forall \mathrm{x}\ \forall \mathrm{y}\ \forall \mathrm{z}\ (\mathrm{R}(\mathrm{x}, \mathrm{y}) \,\&\, \mathrm{R}(\mathrm{y}, \mathrm{z}) \to \mathrm{R}(\mathrm{x}, \mathrm{z})) \to \exists \mathrm{x}\ \forall \mathrm{y}\ \neg \mathrm{R}(\mathrm{x}, \mathrm{y})$$

Рассмотрим такую интерпретацию $\mathcal{I}$:

- Предметная область — множество всех натуральных чисел
- $\overline{R}(a, b) = \mathrm{t} \quad \Leftrightarrow \quad a < b$

Тогда:

- $\mathcal{I} \models \forall \mathrm{x}\ \neg \mathrm{R}(\mathrm{x}, \mathrm{x})$, т.к. никакое число не меньше себя
- $\mathcal{I} \models \forall \mathrm{x}\ \forall \mathrm{y}\ \forall \mathrm{z}\ (\mathrm{R}(\mathrm{x}, \mathrm{y}) \,\&\, \mathrm{R}(\mathrm{y}, \mathrm{z}) \to \mathrm{R}(\mathrm{x}, \mathrm{z}))$,
  т.к. если $a < b$ и $b < c$, то обязательно $a < c$
- $\mathcal{I} \not\models \exists \mathrm{x}\ \forall \mathrm{y}\ \neg \mathrm{R}(\mathrm{x}, \mathrm{y})$,
  т.к. среди натуральных чисел не существует максимального

Следовательно, $\mathcal{I} \not\models \varphi$, и предложение φ **необщезначимо**

Утверждение. **Существует необщезначимое предложение, истинное в любой интерпретации с конечной предметной областью**

Доказательство. Вот пример такого предложения φ:

$$\forall x\, \neg R(x, x) \,\&\, \forall x\, \forall y\, \forall z\, (R(x, y) \,\&\, R(y, z) \to R(x, z)) \to \exists x\, \forall y\, \neg R(x, y)$$

Общее истолкование этой формулы, не зависящее от оценки символа R, можно почерпнуть из теории порядков:

- $\forall x\, \neg R(x, x)$: отношение R антирефлексивно
- $\forall x\, \forall y\, \forall z\, (R(x, y) \,\&\, R(y, z) \to R(x, z))$: отношение R транзитивно
- $\exists x\, \forall y\, \neg R(x, y)$: существует предмет, максимальный относительно R

Если отношение R антирефлексивно и транзитивно, то существует предмет, максимальный относительно R

Иными словами,

Если R — отношение строгого частичного порядка, то существует предмет, максимальный относительно R

Утверждение. **Существует необщезначимое предложение, истинное в любой интерпретации с конечной предметной областью**

Доказательство. Вот пример такого предложения φ:

$$\forall \mathrm{x}\ \neg \mathrm{R}(\mathrm{x}, \mathrm{x}) \,\&\, \forall \mathrm{x}\ \forall \mathrm{y}\ \forall \mathrm{z}\ (\mathrm{R}(\mathrm{x}, \mathrm{y}) \,\&\, \mathrm{R}(\mathrm{y}, \mathrm{z}) \to \mathrm{R}(\mathrm{x}, \mathrm{z})) \to \exists \mathrm{x}\ \forall \mathrm{y}\ \neg \mathrm{R}(\mathrm{x}, \mathrm{y})$$

Если R — отношение строгого частичного порядка, то существует предмет, максимальный относительно R

Как известно (?), в любом **конечном** частично упорядоченном множестве существует максимальный элемент

(А если не известно, то легко доказывается по индукции или получается из более общего утверждения — **леммы Цорна**) ▼

Утверждение. **Существует необщезначимое предложение, истинное в любой интерпретации с конечной предметной областью**

Проверить общезначимость формулы ЛП при помощи
полного перебора интерпретаций оказывается крайне затруднительно:

- Интерпретаций бесконечно много
- Перебирать бесконечные интерпретации проблематично
- Ограничить перебор только конечными интерпретациями невозможно

Чтобы научиться решать эту задачу,
придётся изучить более нетривиальные методы,
и прежде всего изучим самый *идеологически простой* и *универсальный*:

метод семантических таблиц

Математическая логика и логическое программирование

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

Блок 8

Метод семантичесских таблиц:
семантические таблицы

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Чтобы избежать проблем, возникших при попытке адаптировать переборный метод проверки общезначимости формул, попробуем предложить другой метод решения этой задачи, опирающийся **только** на введённую ранее семантику операций

Начнём с примера (назовём его ООФП для отсылок): попробуем обосновать *от противного* общезначимость формулы $\forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \to \mathrm{P}(\mathbf{c})$:

1. Предположим, что $\not\models \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \to \mathrm{P}(\mathbf{c})$
2. Тогда существует интерпретация $\mathcal{I}$, такая что $\mathcal{I} \not\models \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \to \mathrm{P}(\mathbf{c})$
3. Из $\mathcal{I} \not\models \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \to \mathrm{P}(\mathbf{c})$ и семантики $\to$ следует $\mathcal{I} \models \forall \mathrm{x}\, \mathrm{P}(\mathrm{x})$ и $\mathcal{I} \not\models \mathrm{P}(\mathbf{c})$
4. Из $\mathcal{I} \models \forall \mathrm{x}\, \mathrm{P}(\mathrm{x})$ и семантики $\forall$ следует $\mathcal{I} \models \mathrm{P}(\mathbf{c})$
5. Получены соотношения $\mathcal{I} \not\models \mathrm{P}(\mathbf{c})$ и $\mathcal{I} \models \mathrm{P}(\mathbf{c})$ — такого быть не может, а значит, исходное предположение неверно ▼

Попробуем систематизировать ход таких рассуждений и избавиться от лишних слов в доказательстве

Определения

Семантическая таблица (*логики предикатов*) — это упорядоченная пара множеств формул (*логики предикатов*), такая что хотя бы одно из этих множеств непусто

Будем называть два множества семантической таблицы её левой частью (первое) и правой частью (второе)

Будем записывать семантическую таблицу так: $\langle \Gamma \mid \Delta \rangle$, где Γ — левая часть и Δ — правая часть

В записи множеств в семантических таблицах иногда будем опускать фигурные скобки и писать «,» вместо «$\cup$»

Например, $\langle \mathrm{P}(\mathbf{c}), \mathrm{Q}(\mathrm{x}) \mid \exists \mathrm{x}\, \mathrm{Q}(\mathrm{x}) \rangle$ — это семантическая таблица с левой частью $\{\mathrm{P}(\mathbf{c}), \mathrm{Q}(\mathrm{x})\}$ и правой частью $\{\exists \mathrm{x}\, \mathrm{Q}(\mathrm{x})\}$

Содержательно (согласно ООФП), формулу φ в левой части таблицы можно понимать как отвечающую соотношению $\mathcal{I} \models \varphi$, а в правой — как отвечающую соотношению $\mathcal{I} \not\models \varphi$

Определения

Таблица $T = \langle \Gamma \mid \Delta \rangle$ закрыта, если $\Gamma \cap \Delta \neq \emptyset$

Например,

- таблица $\langle \mathrm{P}(\mathbf{c}), \underline{\forall \mathrm{x}\ \mathrm{Q}(\mathrm{x})} \mid \underline{\forall \mathrm{x}\ \mathrm{Q}(\mathrm{x})}, \mathrm{R}(\mathbf{c}) \rangle$ закрыта
- таблица $\langle \mathrm{P}(\mathrm{x}), \neg \mathrm{P}(\mathrm{x}) \mid \mathrm{P}(\mathrm{y}), \mathrm{Q}(\mathrm{x}) \rangle$ незакрыта

Содержательно (согласно ООФП), если $\varphi \in \Gamma \cap \Delta$, то это означает, что получено *явное противоречие*: соотношения $\mathcal{I} \models \varphi$ и $\mathcal{I} \not\models \varphi$

Таблица $T = \langle \Gamma \mid \Delta \rangle$ атомарна, если все формулы из $\Gamma \cup \Delta$ атомарны

Например,

- таблица $\langle \mathrm{P}(\mathrm{x}) \mid \mathrm{Q}(\mathbf{f}(\mathbf{c}), \mathrm{x}), \mathrm{P}(\mathbf{d}) \rangle$ атомарна
- таблицы $\langle \underline{\forall \mathrm{x}}\ \mathrm{P}(\mathrm{x}) \mid \mathrm{Q}(\mathbf{f}(\mathbf{c}), \mathrm{x}), \mathrm{P}(\mathbf{d}) \rangle$ и $\langle \mathrm{P}(\mathrm{x}) \mid \mathrm{Q}(\mathbf{f}(\mathbf{c}), \mathrm{x}) \underline{\vee} \mathrm{P}(\mathbf{d}) \rangle$ неатомарны

Содержательно (согласно ООФП), атомарная таблица отвечает «окончательному» набору соотношений вида $\mathcal{I} \models \varphi$ и $\mathcal{I} \not\models \psi$, из которого невозможно извлечь другие аналогичные соотношения, существенные для ООФП

Определения

Пусть $\widetilde{\mathrm{x}^{\mathrm{n}}}$ — все свободные переменные формул из $\Gamma \cup \Delta$

Таблица $T = \langle \Gamma \mid \Delta \rangle$ выполнима, если существуют интерпретация $\mathcal{I}$ и набор предметов $\widetilde{d^n}$ из области интерпретации, такие что

- $\mathcal{I} \models \varphi(\widetilde{\mathrm{x}^{\mathrm{n}}})[\widetilde{d^n}]$ для любой формулы φ из Γ
- $\mathcal{I} \not\models \psi(\widetilde{\mathrm{x}^{\mathrm{n}}})[\widetilde{d^n}]$ для любой формулы ψ из Δ

Содержательно (согласно ООФП), выполнимость таблицы означает, что в имеющихся соотношениях вида $\mathcal{I} \models \varphi$ и $\mathcal{I} \not\models \psi$ нет противоречия

Определения

Например:

- Таблица $\langle \mathrm{P}(\mathrm{x}) \mid \mathrm{Q}(\mathbf{f}(\mathbf{c}), \mathrm{x}) \rangle$ выполнима (*и незакрыта, и атомарна*)
  Интерпретация и предметы, подтверждающие выполнимость:
  - $\mathsf{D} = \{d\}$, $\overline{\mathrm{P}}(d) = \mathbb{t}$, $\overline{\mathrm{Q}}(d, d) = \mathbb{f}$
  - $d_{\mathrm{x}} = d$

- Таблица $\langle \exists \mathrm{x}\, \mathrm{P}(\mathrm{x}), \neg \mathrm{P}(\mathrm{y}) \mid \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}), \mathrm{P}(\mathrm{x}) \,\&\, \neg \mathrm{P}(\mathrm{x}) \rangle$ выполнима
  (*и незакрыта, и неатомарна*)
  Интерпретация и предметы, подтверждающие выполнимость:
  - $\mathsf{D} = \{0, 1\}$, $\overline{\mathrm{P}}(0) = \mathbb{t}$, $\overline{\mathrm{P}}(1) = \mathbb{f}$
  - $d_{\mathrm{x}} = d_{\mathrm{y}} = 1$

- Таблица $\langle \mathrm{P}(\mathrm{x}) \mid \mathrm{P}(\mathrm{x}), \mathrm{Q}(\mathbf{f}(\mathbf{c}), \mathrm{x}) \rangle$ невыполнима
  (*и закрыта, и атомарна*)

- Таблица $\langle \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \mid \neg \exists \mathrm{x}\, \neg \mathrm{P}(\mathrm{x}) \rangle$ невыполнима
  (*и незакрыта, и неатомарна*)

Основные утверждения

Теорема (о табличной проверке общезначимости)
Для любой формулы φ справедлива равносильность

$$\models \varphi \quad \Leftrightarrow \quad \textbf{таблица } \langle \mid \varphi \rangle \textbf{ невыполнима}$$

Доказательство.

$$\models \varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})$$
$$\Leftrightarrow$$
$$\mathcal{I} \models \varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})[\widetilde{d}^{n}] \text{ для любой интерпретации } \mathcal{I}$$
$$\text{и любого набора предметов } \widetilde{d}^{n}$$
$$\Leftrightarrow$$
$$\text{таблица } \langle \mid \varphi \rangle \text{ невыполнима } \blacktriangledown$$

Утверждение. **Любая закрытая таблица невыполнима**

Утверждение. **Любая незакрытая атомарная таблица выполнима**

Доказательства утверждений опустим для экономии времени

Математическая логика и логическое программирование

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

Блок 9

Подстановки
(основные определения)

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

$\models \varphi \quad \Leftrightarrow$ семантическая таблица $\langle \mid \varphi \rangle$ невыполнима

Для проверки общезначимости формул
достаточно придумать правила преобразования таблиц,
позволяющие извлекать «явные противоречия» (закрытые таблицы)
из таблиц, содержащих «неявные противоречия» (невыполнимых)

Для примера рассмотрим такую невыполнимую таблицу:

$$\langle \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \mid \mathrm{P}(\mathbf{c}) \rangle$$

Чтобы преобразовать эту таблицу в закрытую, достаточно заметить,
что если утверждение $\mathrm{P}(\mathrm{x})$ выполняется для любого предмета x, то оно
выполняется, в частности, и для предмета, обозначенного константой $\mathbf{c}$

Значит, можно подставить на место x константу $\mathbf{c}$
и получить выполнимость утверждения $\mathrm{P}(\mathbf{c})$

Добавив это утверждение в левую часть, получим закрытую таблицу:

$$\langle \forall \mathrm{x}\, \mathrm{P}(\mathrm{x}), \mathrm{P}(\mathbf{c}) \mid \mathrm{P}(\mathbf{c}) \rangle$$

Чтобы строго сформулировать соответствующее правило,
следует строго определить, что такое «подставить»

Подстановки

Пусть заданы множество переменных Var и множество термов Term

Подстановка — это отображение $\theta : \mathrm{Var} \to \mathrm{Term}$

Область подстановки θ: $\mathrm{Dom}_\theta = \{\mathtt{x} \mid \mathtt{x} \in \mathrm{Var}, \theta(\mathtt{x}) \neq \mathtt{x}\}$

Подстановка конечна, если её область конечна

Subst — множество всех конечных подстановок

$\{\mathtt{x}_1/t_1, \ldots, \mathtt{x}_n/t_n\}$ — это конечная подстановка θ, для которой верно:

- $\mathrm{Dom}_\theta = \{\mathtt{x}_1, \ldots, \mathtt{x}_n\}$
- $\theta(\mathtt{x}_i) = t_i, \qquad 1 \leq i \leq n$

Запись $\mathtt{x}/t$, где $\mathtt{x} \in \mathrm{Var}$ и $t \in \mathrm{Term}$, называется связкой

Содержательно, связка $\mathtt{x}/t$ означает, что при *применении* (*выполнении*) подстановки переменная $\mathtt{x}$ должна быть заменена на терм t

ε — это тождественная (пустая) подстановка: $\mathrm{Dom}_\varepsilon = \emptyset$

Подстановки

Пусть E — логическое выражение логики предикатов (терм или формула) и θ — подстановка

Результат $E\theta$ применения подстановки θ к E определяется так:

$\mathrm{x}\theta = \theta(\mathrm{x})$	$(\mathrm{x} \in \mathrm{Var})$
$\mathbf{c}\theta = \mathbf{c}$	$(\mathbf{c} \in \mathrm{Const})$
$\mathbf{f}(t_1, \ldots, t_n)\theta = \mathbf{f}(t_1\theta, \ldots, t_n\theta)$	$(\mathbf{f} \in \mathrm{Func},\ t_1, \ldots, t_n \in \mathrm{Term})$
$\mathrm{P}(t_1, \ldots, t_n)\theta = \mathrm{P}(t_1\theta, \ldots, t_n\theta)$	$(\mathrm{P} \in \mathrm{Pred})$
$(\varphi \,\&\, \psi)\theta = (\varphi\theta \,\&\, \psi\theta)$	$(\varphi, \psi \in \mathrm{Form})$
$(\varphi \vee \psi)\theta = (\varphi\theta \vee \psi\theta)$	
$(\varphi \to \psi)\theta = (\varphi\theta \to \psi\theta)$	
$(\neg\varphi)\theta = (\neg\varphi\theta)$	
$(\forall \mathrm{x}\ \ \varphi)\theta = (\forall \mathrm{x}\ \ \varphi\theta')$	$(\theta'(\mathrm{x}) = \mathrm{x};$
$(\exists \mathrm{x}\ \ \varphi)\theta = (\exists \mathrm{x}\ \ \varphi\theta')$	$\theta'(\mathrm{y}) = \theta(\mathrm{y})$, если $\mathrm{y} \neq \mathrm{x})$

Иными словами, $E\theta$ получается из выражения E так:

- E — терм $\Rightarrow$ все вхождения переменных заменяются на их θ-образы
- E — формула $\Rightarrow$ все **свободные** вхождения переменных заменяются на их θ-образы

Подстановки

Пример применения подстановки к формуле

$\varphi = \forall \mathrm{x}\ (\mathrm{P}(\mathrm{x}) \to \neg \mathrm{R}(\mathrm{y})) \to \mathrm{R}(\mathbf{f}(\mathrm{x})) \vee \exists y\ \mathrm{P}(\mathrm{y}) \vee \mathrm{R}(\mathrm{u})$
$\theta = \{\mathrm{x}/\mathbf{g}(\mathrm{x}, \mathbf{c}), \mathrm{y}/\mathrm{x}, \mathrm{z}/\mathbf{f}(\mathrm{z})\}$

Выделяются все свободные вхождения переменных в φ

$\forall \mathrm{x}\ (\mathrm{P}(\mathrm{x}) \to \neg \mathrm{R}(\mathrm{y})) \to \mathrm{R}(\mathbf{f}(\mathrm{x})) \vee \exists \mathrm{y}\ \mathrm{P}(\mathrm{y}) \vee \mathrm{R}(\mathrm{u})$

Все выделенные вхождения заменяются согласно θ

$\varphi\theta = \forall \mathrm{x}\ (\mathrm{P}(\mathrm{x}) \to \neg \mathrm{R}(\mathrm{x})) \to \mathrm{R}(\mathbf{f}(\mathbf{g}(\mathrm{x}, \mathbf{c}))) \vee \exists \mathrm{y}\ \mathrm{P}(\mathrm{y}) \vee \mathrm{R}(\mathrm{u})$

Подстановки

При применении подстановок для выделения «частных случаев» следует соблюдать осторожность

Например:

$\varphi(\mathrm{x}) = \forall \mathrm{x}\ \exists \mathrm{y}\ \mathrm{P}(\mathrm{x}, \mathrm{y}) \to \exists y\ \mathrm{P}(\mathrm{x}, \mathrm{y})$

«если у каждого есть дед, то у x тоже есть дед»

Очевидно, что $\models \varphi(\mathrm{x})$

$\varphi\{\mathrm{x}/\mathrm{y}\} = \forall \mathrm{x}\ \exists \mathrm{y}\ \mathrm{P}(\mathrm{x}, \mathrm{y}) \to \exists \mathrm{y}\ \mathrm{P}(\mathrm{y}, \mathrm{y})$

«если у каждого есть дед, то есть и тот, кто сам себе дед»

Очевидно, что $\not\models \varphi\theta$

Почему смысл формулы после применения подстановки так исказился?

Подстановки

Переменная x свободна для терма t в формуле φ, если
ни одно свободное вхождение переменной x не лежит
в областях действия кванторов, связывающих переменные из Var_t

Подстановка $\theta = \{\mathrm{x}_1/t_1, \ldots, \mathrm{x}_n/t_n\}$ — правильная для формулы φ,
если для каждой связки x_i/t_i
переменная x_i свободна для терма t_i в формуле φ

Например, для формулы $\forall \mathrm{x}\, \exists \mathrm{y}\, \mathrm{P}(\mathrm{x}, \mathrm{y}) \to \exists \mathrm{y}\, \mathrm{P}(\mathrm{x}, \mathrm{y})$

- подстановка $\{\mathrm{x}/\mathbf{f}(\mathrm{u}, \mathrm{v})\}$ — правильная:
  все вхождения u и v в подставляемый терм свободны
- подстановка $\{\mathrm{x}/\mathrm{y}\}$ — неправильная:
  вхождение y в подставляемый терм оказывается связанным

Подстановки

Утверждение (о правильной подстановке)

Для любых формулы $\varphi(\widetilde{\mathrm{x}}^{\mathrm{n}}, \mathrm{x})$, интерпретации $\mathcal{I}$, набора предметов $\widetilde{d}^n$ и подстановки $\{\mathrm{x}/t(\widetilde{\mathrm{x}}^{\mathrm{n}})\}$, правильной для φ, верно:

$$\mathcal{I} \models \varphi[\widetilde{d}^n, t[\widetilde{d}^n]] \quad \Leftrightarrow \quad \mathcal{I} \models \varphi\{\mathrm{x}/t\}[\widetilde{d}^n]$$

Доказательство опустим для экономии времени

и ограничимся небольшим **примером**:

Пусть $\mathbf{1}, \mathbf{3} \in$ Const, $=^{(2)} \in$ Pred, $+^{(2)} \in$ Func

и все эти символы имеют «естественную» арифметическую оценку в $\mathcal{I}$: $\overline{\mathbf{1}} = 1$, $\overline{\mathbf{3}} = 3$, $\equiv$ и $\overline{+}$ — отношение равенства и операция сложения чисел

Тогда верно следующее:

$$\mathcal{I} \models (\mathrm{x} = \mathbf{3})[\mathrm{y}/2, \mathrm{x}/3]$$

$$\Leftrightarrow \qquad (\overline{\mathbf{3}} = 3 = (\mathbf{1} + \mathrm{y})[\mathrm{y}/2])$$

$$\mathcal{I} \models (\mathrm{x} = \mathbf{3})[\mathrm{y}/2, \mathrm{x}/((\mathbf{1} + \mathrm{y})[\mathrm{y}/2])]$$

$$\Leftrightarrow \qquad (\text{утверждение выше})$$

$$\mathcal{I} \models (\mathbf{1} + \mathrm{y} = \mathbf{3})[\mathrm{y}/2]$$

Математическая логика и логическое программирование

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

Блок 10

Метод семантических таблиц:
табличный вывод

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

$\models \varphi \quad \Leftrightarrow$ семантическая таблица $\langle \mid \varphi \rangle$ невыполнима

Для доказательства общезначимости формул
(и более широко — невыполнимости таблиц)
будем применять правила заранее сформулированного списка

Доказательства такого вида:
преобразование записей согласно заданному своду правил —
принято называть логическим выводом

Логический вывод, в котором преобразуются семантические таблицы,
принято называть табличным выводом, и соответствующие
правила преобразования — правилами табличного вывода

Начнём с определения свода этих правил

Правила табличного вывода

Будем использовать правила табличного вывода двух видов:

$$(*)\colon \frac{T_0}{T_1}, \qquad (**)\colon \frac{T_0}{T_1,\ T_2},$$

где T_0, T_1, T_2 — семантические таблицы

Согласно правилу, рассматриваемая таблица T_0 преобразуется

- $(*)$ в таблицу T_1 для последующего рассмотрения
- $(**)$ в таблицы T_1 и T_2 для поочерёдного рассмотрения

При этом правила будут подобраны так, чтобы

- $(*)$ таблица T_0 была выполнима тогда и только тогда, когда и T_1
- $(**)$ таблица T_0 была выполнима тогда и только тогда, когда и **хотя бы одна** из таблиц T_1, T_2

Таблицы T_1, T_2 под чертой в правилах иногда называют альтернативами

Правила табличного вывода

Включим в свод 12 правил табличного вывода:

- $12 = 2 \cdot 6$
  - 2 части таблицы: левая, правая
  - 6 логических операций: $\&$, $\vee$, $\to$, $\neg$, $\forall$, $\exists$
- согласно каждому правилу, в одной из частей таблицы выбирается одна формула, и эта формула преобразуется в одну или несколько в зависимости от её вида и расположения

В правилах будут использоваться следующие обозначения:

- Γ, Δ — произвольные множества формул
- φ, ψ — произвольные формулы
- x — произвольная предметная переменная
- t — произвольный терм, такой что подстановка $\{\mathrm{x}/t\}$ правильна для φ
- $\mathbf{c}$ — произвольная константа, не содержащаяся в формулах из $\Gamma \cup \Delta \cup \{\varphi\}$

Правила табличного вывода

$$L\&: \frac{\langle \Gamma, \varphi \,\&\, \psi \mid \Delta \rangle}{\langle \Gamma, \varphi, \psi \mid \Delta \rangle} \qquad R\&: \frac{\langle \Gamma \mid \Delta, \varphi \,\&\, \psi \rangle}{\langle \Gamma \mid \Delta, \varphi \rangle, \langle \Gamma \mid \Delta, \psi \rangle}$$

$$L\vee: \frac{\langle \Gamma, \varphi \vee \psi \mid \Delta \rangle}{\langle \Gamma, \varphi \mid \Delta \rangle, \langle \Gamma, \psi \mid \Delta \rangle} \qquad R\vee: \frac{\langle \Gamma \mid \Delta, \varphi \vee \psi \rangle}{\langle \Gamma \mid \Delta, \varphi, \psi \rangle}$$

$$L\to: \frac{\langle \Gamma, \varphi \to \psi \mid \Delta \rangle}{\langle \Gamma, \psi \mid \Delta \rangle, \langle \Gamma \mid \Delta, \varphi \rangle} \qquad R\to: \frac{\langle \Gamma \mid \Delta, \varphi \to \psi \rangle}{\langle \Gamma, \varphi \mid \Delta, \psi \rangle}$$

$$L\neg: \frac{\langle \Gamma, \neg\varphi \mid \Delta \rangle}{\langle \Gamma \mid \Delta, \varphi \rangle} \qquad R\neg: \frac{\langle \Gamma \mid \Delta, \neg\varphi \rangle}{\langle \Gamma, \varphi \mid \Delta \rangle}$$

$$L\forall: \frac{\langle \Gamma, \forall \mathrm{x}\ \varphi \mid \Delta \rangle}{\langle \Gamma, \forall \mathrm{x}\ \varphi, \varphi\{\mathrm{x}/t\} \mid \Delta \rangle} \qquad R\forall: \frac{\langle \Gamma \mid \Delta, \forall \mathrm{x}\ \varphi \rangle}{\langle \Gamma \mid \Delta, \varphi\{\mathrm{x}/\mathbf{c}\} \rangle}$$

$$L\exists: \frac{\langle \Gamma, \exists \mathrm{x}\ \varphi \mid \Delta \rangle}{\langle \Gamma, \varphi\{\mathrm{x}/\mathbf{c}\} \mid \Delta \rangle} \qquad R\exists: \frac{\langle \Gamma \mid \Delta, \exists \mathrm{x}\ \varphi \rangle}{\langle \Gamma \mid \Delta, \exists \mathrm{x}\ \varphi, \varphi\{\mathrm{x}/t\} \rangle}$$

Правила табличного вывода

Пара слов об ограничениях
на терм t и константу c в правилах $L\forall$, $R\forall$, $L\exists$, $R\exists$

Если разрешить подставлять любые термы в $L\forall$, $R\exists$:

$$\frac{\langle \forall \mathrm{x}\ \exists \mathrm{y}\ \mathrm{P}(\mathrm{x}, \mathrm{y}) \mid \exists \mathrm{y}\ \mathrm{P}(\mathrm{y}, \mathrm{y}) \rangle}{\langle \forall \mathrm{x}\ \exists \mathrm{y}\ \mathrm{P}(\mathrm{x}, \mathrm{y}), \exists \mathrm{y}\ \mathrm{P}(\mathrm{y}, \mathrm{y}) \mid \exists \mathrm{y}\ \mathrm{P}(\mathrm{y}, \mathrm{y}) \rangle}$$

— выполнимая таблица

— невыполнимая таблица

Если разрешить подставлять «использованные» константы в $L\exists$, $R\forall$:

$$\frac{\langle \exists \mathrm{x}\ \mathrm{P}(\mathrm{x}) \mid \mathrm{P}(\mathbf{c}) \rangle}{\langle \mathrm{P}(\mathbf{c}) \mid \mathrm{P}(\mathbf{c}) \rangle}$$

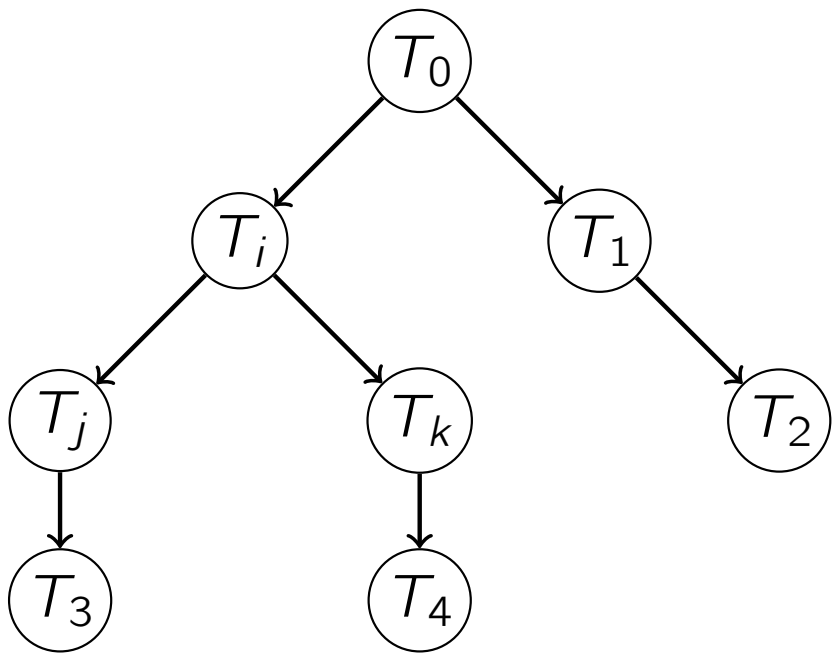
— выполнимая таблица

— невыполнимая таблица

Табличный вывод

Табличный вывод для таблицы T_0 — это размеченное корневое ориентированное дерево следующего вида:

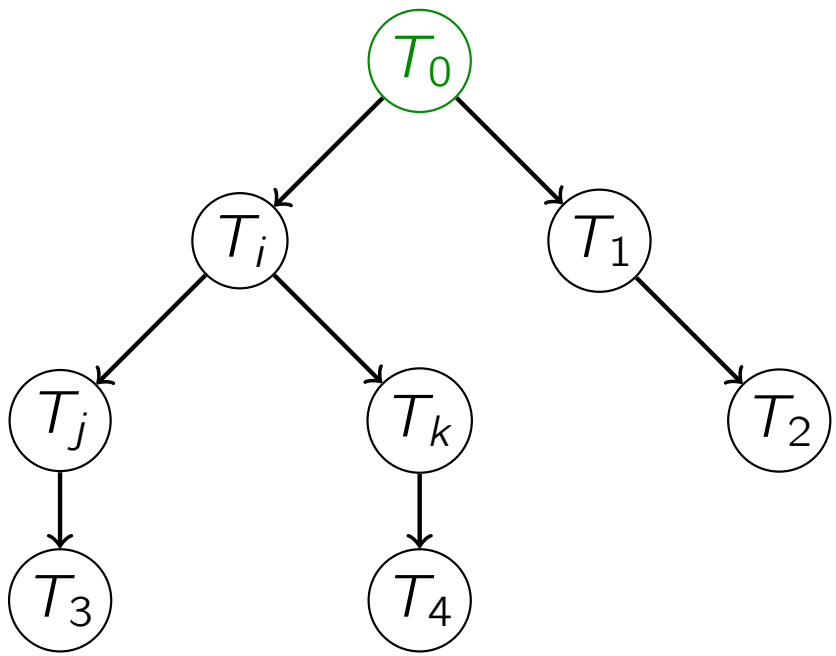
1. Всем вершинам приписаны семантические таблицы



Табличный вывод

Табличный вывод для таблицы T_0 — это
размеченное корневое ориентированное дерево следующего вида:

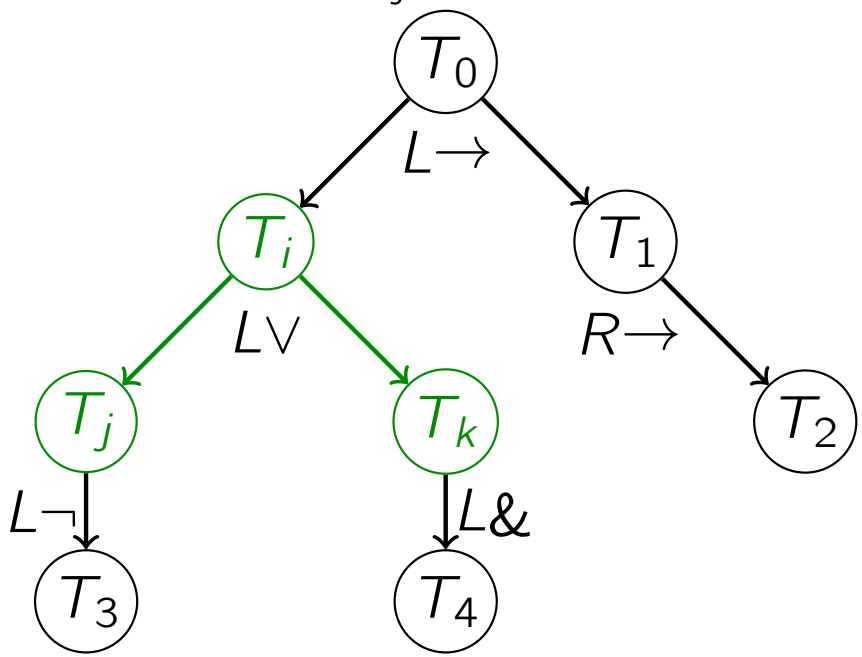
2. Корню приписана таблица T_0



Табличный вывод

Табличный вывод для таблицы T_0 — это
размеченное корневое ориентированное дерево следующего вида:

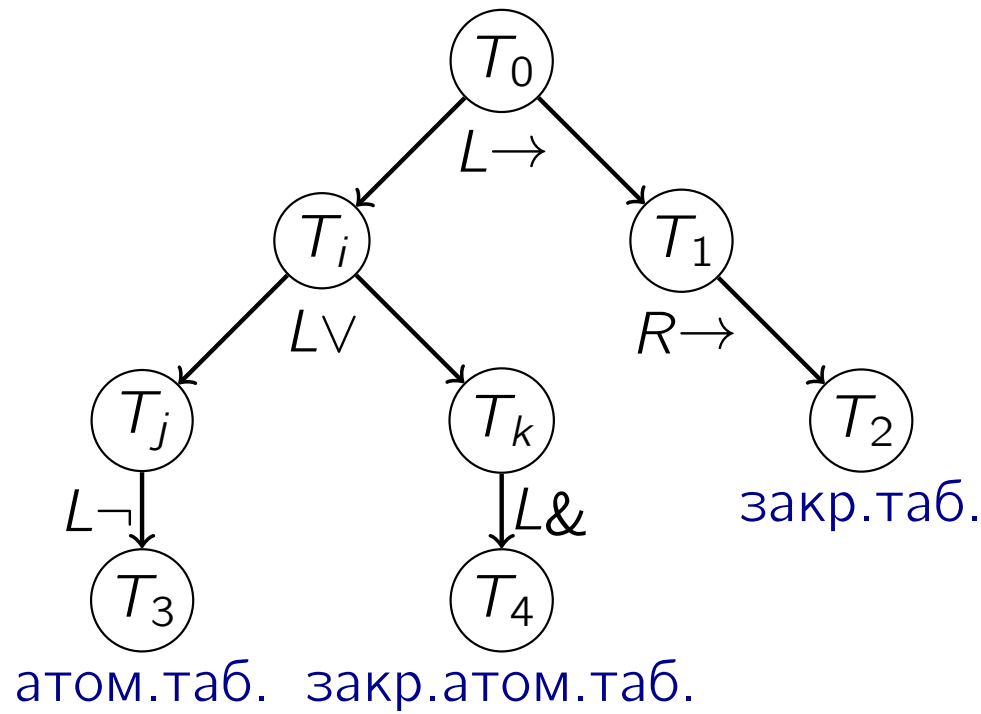
3. Из каждой вершины (T_i) исходит не более двух дуг, и если исходит
   - ровно одна дуга (в (T_j)), то $\dfrac{T_i}{T_j}$ — правило табличного вывода
   - две дуги (в (T_j), (T_k)), то $\dfrac{T_i}{T_j,\ T_k}$ — правило табличного вывода



Табличный вывод

Табличный вывод для таблицы T_0 — это
размеченное корневое ориентированное дерево следующего вида:

4. все таблицы, приписанные листьям, закрыты или атомарны
   (в том числе могут быть одновременно закрытыми и атомарными)



Табличный вывод

Табличный вывод успешен, если
он конечен и всем его листьям приписаны закрытые таблицы

Успешный табличный вывод явно демонстрирует, что таблица,
для которой он построен, невыполнима (*докажем это позже*)

В частности, согласно теореме о табличной проверке общезначимости,
если этот вывод построен для таблицы $\langle \mid \varphi \rangle$, то верно $\models \varphi$

Перед строгой формулировкой и обоснованием
свойств успешных выводов приведём несколько примеров

Примеры табличных выводов

Вывод успешен

При этом для любых формул φ, ψ верно $\models (\varphi \to \neg\psi) \to (\psi \to \neg\varphi)$

Примеры табличных выводов

$$\langle \mid \forall x\,(A(x) \to B(x)) \to (\forall x\,A(x) \to \forall x\,B(x)) \rangle$$

$\downarrow R\to$

$$\langle \forall x\,(A(x) \to B(x)) \mid \forall x\,A(x) \to \forall x\,B(x) \rangle$$

$\downarrow R\to$

$$\langle \forall x\,(A(x) \to B(x)), \forall x\,A(x) \mid \forall x\,B(x) \rangle$$

$\downarrow R\forall$

$$\langle \forall x\,(A(x) \to B(x)), \forall x\,A(x) \mid B(\mathbf{c}) \rangle$$

$\downarrow L\forall$

$$\langle \forall x\,(A(x) \to B(x)), \forall x\,A(x), A(\mathbf{c}) \mid B(\mathbf{c}) \rangle$$

$\downarrow L\forall$

$$\langle \forall x\,(A(x) \to B(x)), \forall x\,A(x), A(\mathbf{c}) \to B(\mathbf{c}), A(\mathbf{c}) \mid B(\mathbf{c}) \rangle$$

$\swarrow \quad L\to \quad \searrow$

$$\langle \begin{matrix} \forall x\,(A(x) \to B(x)), \\ \forall x\,A(x), B(\mathbf{c}), A(\mathbf{c}) \end{matrix} \mid B(\mathbf{c}) \rangle \qquad \langle \begin{matrix} \forall x\,(A(x) \to B(x)), \\ \forall x\,A(x), A(\mathbf{c}) \end{matrix} \mid B(\mathbf{c}), A(\mathbf{c}) \rangle$$

Закрытая таблица — Закрытая таблица

Вывод успешен

При этом $\models \forall x\,(A(x) \to B(x)) \to (\forall x\,A(x) \to \forall x\,B(x))$

Примеры табличных выводов

$$\langle \mid \exists x\, P(x) \to \forall x\, P(x) \rangle$$
$$\downarrow R\to$$
$$\langle \exists x\, P(x) \mid \forall x\, P(x) \rangle$$
$$\downarrow L\exists$$
$$\langle P(\mathbf{c_1}) \mid \forall x\, P(x) \rangle$$
$$\downarrow R\forall$$
$$\langle P(\mathbf{c_1}) \mid P(\mathbf{c_2}) \rangle$$

Незакрытая атомарная таблица

Вывод неуспешен

При этом $\not\models \exists x\, P(x) \to \forall x\, P(x)$

Примеры табличных выводов

$$\langle \mid \forall x\ \exists y\ P(x,y) \to \exists y\ \forall x\ P(x,y) \rangle$$
$$\downarrow R\to$$
$$\langle \forall x\ \exists y\ P(x,y) \mid \exists y\ \forall x\ P(x,y) \rangle$$
$$\downarrow L\forall$$
$$\langle \forall x\ \exists y\ P(x,y), \exists y\ P(\mathbf{c_1},y) \mid \exists y\ \forall x\ P(x,y) \rangle$$
$$\downarrow R\exists$$
$$\langle \forall x\ \exists y\ P(x,y), \exists y\ P(\mathbf{c_1},y) \mid \exists y\ \forall x\ P(x,y), \forall x\ P(x,\mathbf{c_2}) \rangle$$
$$\downarrow L\exists$$
$$\langle \forall x\ \exists y\ P(x,y), P(\mathbf{c_1},\mathbf{c_3}) \mid \exists y\ \forall x\ P(x,y), \forall x\ P(x,\mathbf{c_2}) \rangle$$
$$\downarrow R\forall$$
$$\langle \forall x\ \exists y\ P(x,y), P(\mathbf{c_1},\mathbf{c_3}) \mid \exists y\ \forall x\ P(x,y), P(\mathbf{c_4},\mathbf{c_2}) \rangle$$
$$\downarrow L\forall$$
$$\infty$$

Вывод бесконечен (и, следовательно, неуспешен)

При этом $\not\models \forall x\ \exists y\ P(x,y) \to \exists y\ \forall x\ P(x,y)$

Примеры табличных выводов

$$\langle \mid \exists x\, \forall y\, P(x,y) \to \forall y\, \exists x\, P(x,y) \rangle$$
$$\downarrow R\to$$
$$\langle \exists x\, \forall y\, P(x,y) \mid \forall y\, \exists x\, P(x,y) \rangle$$
$$\downarrow L\exists$$
$$\langle \forall y\, P(\mathbf{c_1},y) \mid \forall y\, \exists x\, P(x,y) \rangle$$
$$\downarrow R\forall$$
$$\langle \forall y\, P(\mathbf{c_1},y) \mid \exists x\, P(x,\mathbf{c_2}) \rangle$$
$$\downarrow L\forall$$
$$\langle \forall y\, P(\mathbf{c_1},y), P(\mathbf{c_1},\mathbf{c_3}) \mid \exists x\, P(x,\mathbf{c_2}) \rangle$$
$$\downarrow R\exists$$
$$\langle \forall y\, P(\mathbf{c_1},y), P(\mathbf{c_1},\mathbf{c_3}) \mid \exists x\, P(x,\mathbf{c_2}), P(\mathbf{c_4},\mathbf{c_2}) \rangle$$
$$\downarrow L\forall$$
$$\infty$$

Вывод бесконечен (и, следовательно, неуспешен)

При этом $\not\models \exists x\, \forall y\, P(x,y) \to \forall y\, \exists x\, P(x,y)$

Математическая логика и логическое программирование

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

Блок 11

Метод семантических таблиц:
корректность табличного вывода

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

ВМК МГУ, 2023/2024, осенний семестр

Лемма (о корректности правил табличного вывода)
Для любого правила табличного вывода $\frac{T_0}{T_1(,\, T_2)}$**:**

$$L\&,\ R\&,\ L\vee,\ R\vee,\ L\to,\ R\to,\ L\neg,\ R\neg,\ L\forall,\ R\forall,\ L\exists,\ R\exists$$ **—**

таблица T_0 **выполнима тогда и только тогда,**
когда выполнима таблица T_1 **(или выполнима таблица** T_2**)**

Доказательство

$L\to$: $\dfrac{\langle \Gamma, \varphi \to \psi \mid \Delta \rangle}{\langle \Gamma, \psi \mid \Delta \rangle, \langle \Gamma \mid \Delta, \varphi \rangle}$

($\Rightarrow$) Пусть верхняя таблица выполнима —
покажем, что тогда выполнима хотя бы одна из нижних таблиц

По определению выполнимости таблицы,
существуют интерпретация $\mathcal{I}$ и набор предметов $\widetilde{d}^n$, такие что

- для любой формулы χ' из Γ верно $\mathcal{I} \models \chi'[\widetilde{d}^n]$
- для любой формулы χ'' из Δ верно $\mathcal{I} \not\models \chi''[\widetilde{d}^n]$
- $\mathcal{I} \models (\varphi \to \psi)[\widetilde{d}^n]$

Лемма (о корректности правил табличного вывода)
Для любого правила табличного вывода $\frac{T_0}{T_1(,\, T_2)}$:

$$L\&,\ R\&,\ L\vee,\ R\vee,\ L\to,\ R\to,\ L\neg,\ R\neg,\ L\forall,\ R\forall,\ L\exists,\ R\exists$$ **—**

таблица T_0 выполнима тогда и только тогда,
когда выполнима таблица T_1 (или выполнима таблица T_2)

Доказательство

$(L\to\ (\Rightarrow)\qquad \chi' \in \Gamma \Rightarrow \mathcal{I} \models \chi'[\widetilde{d}^n] \qquad \chi'' \in \Delta \Rightarrow \mathcal{I} \not\models \chi''[\widetilde{d}^n] \qquad \mathcal{I} \models (\varphi \to \psi)[\widetilde{d}^n])$

Так как $\mathcal{I} \models (\varphi \to \psi)[\widetilde{d}^n]$, то, согласно семантике $\to$,
верно хотя бы одно из двух:

- $\mathcal{I} \not\models \varphi[\widetilde{d}^n]$
- $\mathcal{I} \models \psi[\widetilde{d}^n]$

Следовательно, хотя бы одна из таблиц $\langle \Gamma, \psi \mid \Delta \rangle$, $\langle \Gamma \mid \Delta, \varphi \rangle$ выполнима

($\Leftarrow$) Рассуждения аналогичны

$L\&,\ R\&,\ L\vee,\ R\vee,\ R\to,\ L\neg,\ R\neg$ — рассуждения аналогичны

Лемма (о корректности правил табличного вывода)
Для любого правила табличного вывода $\frac{T_0}{T_1(,\, T_2)}$**:**

$$L\&,\ R\&,\ L\vee,\ R\vee,\ L\to,\ R\to,\ L\neg,\ R\neg,\ L\forall,\ R\forall,\ L\exists,\ R\exists$$ —

таблица T_0 **выполнима тогда и только тогда,**
когда выполнима таблица T_1 **(или выполнима таблица** T_2**)**

Доказательство. $L\forall$: $\frac{\langle \Gamma, \forall \mathrm{x}\, \varphi \mid \Delta\rangle}{\langle \Gamma, \forall \mathrm{x}\, \varphi, \varphi\{\mathrm{x}/t\} \mid \Delta\rangle}$

($\Leftarrow$) *Очевидно, что* если вычеркнуть «$\varphi\{\mathrm{x}/t\}$» из выполнимой (нижней) таблицы, то она (станет верхней и) останется выполнимой

($\Rightarrow$) Пусть верхняя таблица выполнима,
и пусть $\widetilde{\mathrm{x}}^{\mathrm{n}}$ — все свободные переменные формул нижней таблицы
Верхняя таблица выполнима $\Leftrightarrow$
существуют интерпретация $\mathcal{I}$ и набор предметов $\widetilde{d}^{\,n}$, такие что

- $\mathcal{I} \models \psi(\widetilde{\mathrm{x}}^{\mathrm{n}})[\widetilde{d}^{\,n}]$ для каждой формулы ψ из Γ
- $\mathcal{I} \not\models \chi(\widetilde{\mathrm{x}}^{\mathrm{n}})[\widetilde{d}^{\,n}]$ для каждой формулы χ из Δ
- $\mathcal{I} \models (\forall \mathrm{x}\, \varphi)[\widetilde{d}^{\,n}]$

При этом $\mathcal{I} \models (\forall \mathrm{x}\, \varphi)[\widetilde{d}^{\,n}] \ \Rightarrow\ \mathcal{I} \models \varphi[t[\widetilde{d}^{\,n}], \widetilde{d}^{\,n}] \ \Rightarrow\ \mathcal{I} \models \varphi\{\mathrm{x}/t\}[\widetilde{d}^{\,n}]$
Значит, нижняя таблица также выполнима

А где используется **правильность** подстановки $\{\mathrm{x}/t\}$ для φ?

Лемма (о корректности правил табличного вывода)
Для любого правила табличного вывода $\frac{T_0}{T_1(,\, T_2)}$:

$$L\&,\ R\&,\ L\vee,\ R\vee,\ L\to,\ R\to,\ L\neg,\ R\neg,\ L\forall,\ R\forall,\ L\exists,\ R\exists$$

— таблица T_0 выполнима тогда и только тогда,
когда выполнима таблица T_1 (или выполнима таблица T_2)

Доказательство. *L*∃: $\frac{\langle \Gamma, \exists \mathrm{x}\, \varphi \mid \Delta\rangle}{\langle \Gamma, \varphi\{\mathrm{x}/\mathbf{c}\} \mid \Delta\rangle}$

($\Leftarrow$) *Очевидно*, т.к. если $\mathcal{I} \models \varphi\{\mathrm{x}/\mathbf{c}\}[\widetilde{d}^n]$, то $\mathcal{I} \models (\exists \mathrm{x}\, \varphi)[\widetilde{d}^n]$

($\Rightarrow$) Пусть верхняя таблица выполнима,
и $\widetilde{\mathrm{x}}^{\mathrm{n}}$ — все свободные переменные формул верхней таблицы

Верхняя таблица выполнима $\Leftrightarrow$
существуют интерпретация $\mathcal{I}$ и набор предметов $\widetilde{d}^n$, такие что

- $\mathcal{I} \models \psi(\widetilde{\mathrm{x}}^{\mathrm{n}})[\widetilde{d}^n]$ для каждой формулы ψ из Γ
- $\mathcal{I} \not\models \chi(\widetilde{\mathrm{x}}^{\mathrm{n}})[\widetilde{d}^n]$ для каждой формулы χ из Δ
- $\mathcal{I} \models (\exists \mathrm{x}_0\, \varphi)[\widetilde{d}^n]$ — а значит, существует предмет d_0, такой что $\mathcal{I} \models \varphi[d_0, \widetilde{d}^n]$

Лемма (о корректности правил табличного вывода)
Для любого правила табличного вывода $\frac{T_0}{T_1(,\, T_2)}$**:**

$$L\&,\ R\&,\ L\vee,\ R\vee,\ L\to,\ R\to,\ L\neg,\ R\neg,\ L\forall,\ R\forall,\ L\exists,\ R\exists$$ **—**

таблица T_0 **выполнима тогда и только тогда,**
когда выполнима таблица T_1 **(или выполнима таблица** T_2**)**

Доказательство. $L\exists$: $\frac{\langle \Gamma, \exists \mathrm{x}\, \varphi \mid \Delta\rangle}{\langle \Gamma, \varphi\{\mathrm{x}/\mathbf{c}\} \mid \Delta\rangle}$

$(\Rightarrow)$ $\quad \psi \in \Gamma \Rightarrow \mathcal{I} \models \psi[\widetilde{d}^n] \quad \chi \in \Delta \Rightarrow \mathcal{I} \not\models \psi[\widetilde{d}^n] \quad \mathcal{I} \models \varphi[d_0, \widetilde{d}^n]$

Рассмотрим интерпретацию $\mathcal{J}$,
отличающуюся от $\mathcal{I}$ только оценкой константы $\mathbf{c}$: $\overline{\mathbf{c}} = d_0$

Тогда $\mathcal{J} \models (\varphi\{\mathrm{x}/\mathbf{c}\})[\widetilde{d}^n]$

Кроме того,

- $\mathcal{J} \models \psi[\widetilde{d}^n]$ для каждой формулы ψ из Γ
- $\mathcal{J} \not\models \chi[\widetilde{d}^n]$ для каждой формулы χ из Δ

Значит, нижняя таблица выполнима

А где используется тот факт, что $\mathbf{c}$ — «новая» константа?

$R\forall$, $R\exists$ — рассуждения аналогичны ▼

Теорема (о корректности табличного вывода)
Если для семантической таблицы T существует успешный табличный вывод, то таблица T невыполнима

Доказательство.
Следует из определения успешного табличного вывода, утверждения о невыполнимости закрытых таблиц и леммы о корректности правил табличного вывода ▼

Следствие
Если для таблицы $\langle \mid \varphi \rangle$ существует успешный табличный вывод, то $\models \varphi$

А верно ли утверждение в обратную сторону?

Математическая логика и логическое программирование

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

Блок 12

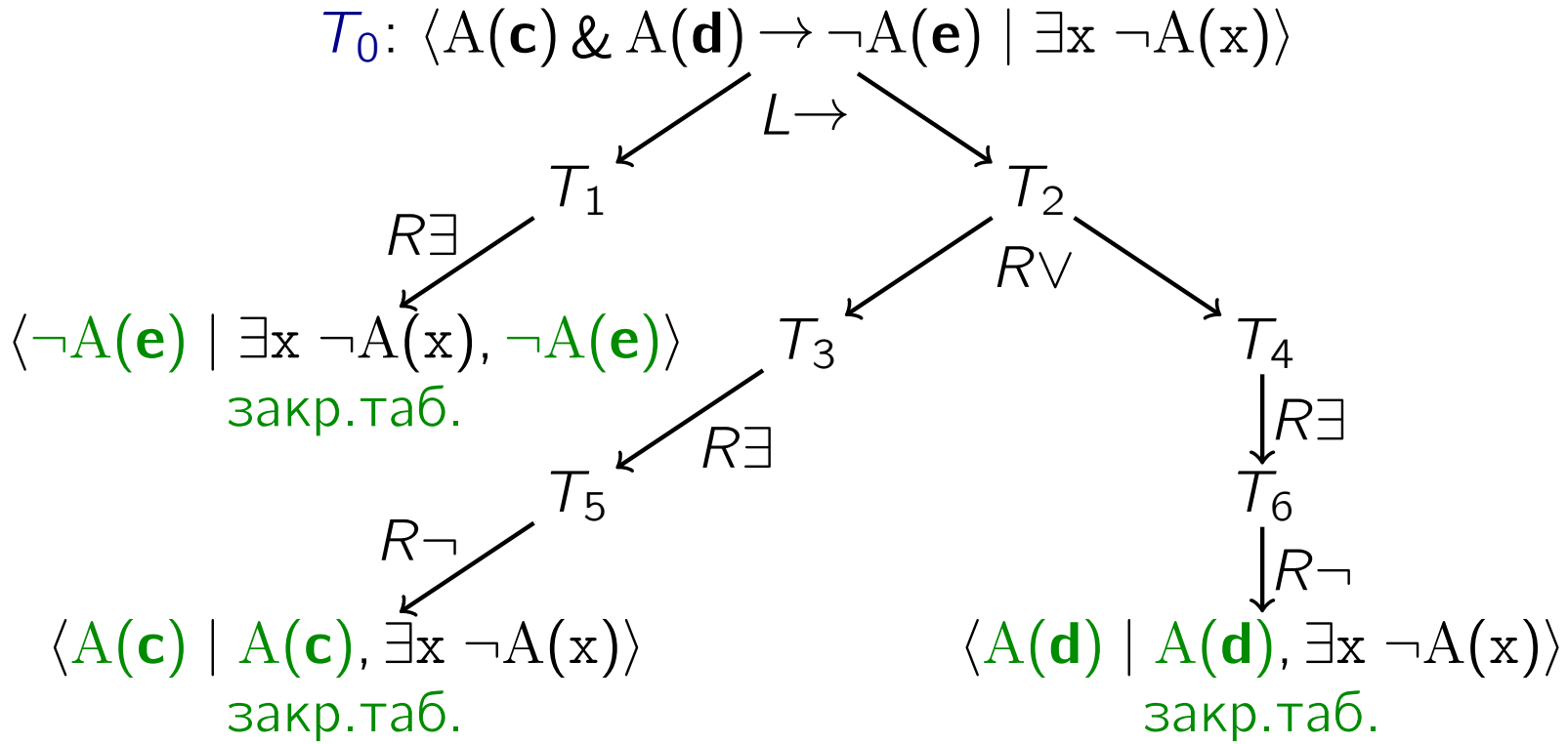
Метод семантических таблиц:
полнота табличного вывода

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

ВМК МГУ, 2023/2024, осенний семестр

Напоминание

Корректность табличного вывода:
если для таблицы T_0 существует успешный табличный вывод,
то таблица T_0 невыполнима



А верно ли утверждение в обратную сторону?
(*если таблица невыполнима, то для неё существует успешный вывод*)

Теорема (о полноте табличного вывода)
Для любой невыполнимой семантической таблицы существует успешный табличный вывод

Доказательство.

Рассмотрим произвольную невыполнимую семантическую таблицу

$$T_0 = \langle \Gamma_0 \mid \Delta_0 \rangle$$

Покажем, как можно **построить** успешный вывод $\mathfrak{D}$ для T_0

Для простоты обсудим частный случай с такими ограничениями:

- множества Γ_0, Δ_0 конечны
- все формулы из Γ_0, Δ_0 замкнуты
- в сигнатуре нет ни одного функционального символа

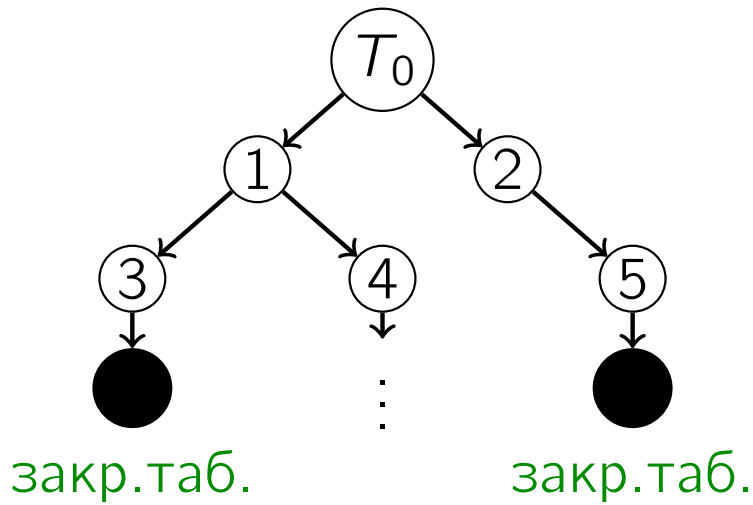
Сформулируем **стратегию** построения вывода,
которой (*как будет показано*) достаточно придерживаться
для построения требуемого вывода $\mathfrak{D}$, начиная с исходной таблицы T_0

Доказательство теоремы о полноте табличного вывода.

Включим в стратегию построения успешного вывода три ограничения, разрешив выбирать остальные детали построения произвольно:

1. Дерево вывода строится при помощи обхода в ширину

То есть правила применяются к незакрытым неатомарным таблицам в порядке появления этих таблиц в построенном фрагменте дерева



Тогда каждая таблица каждой ветви вывода рано или поздно будет построена

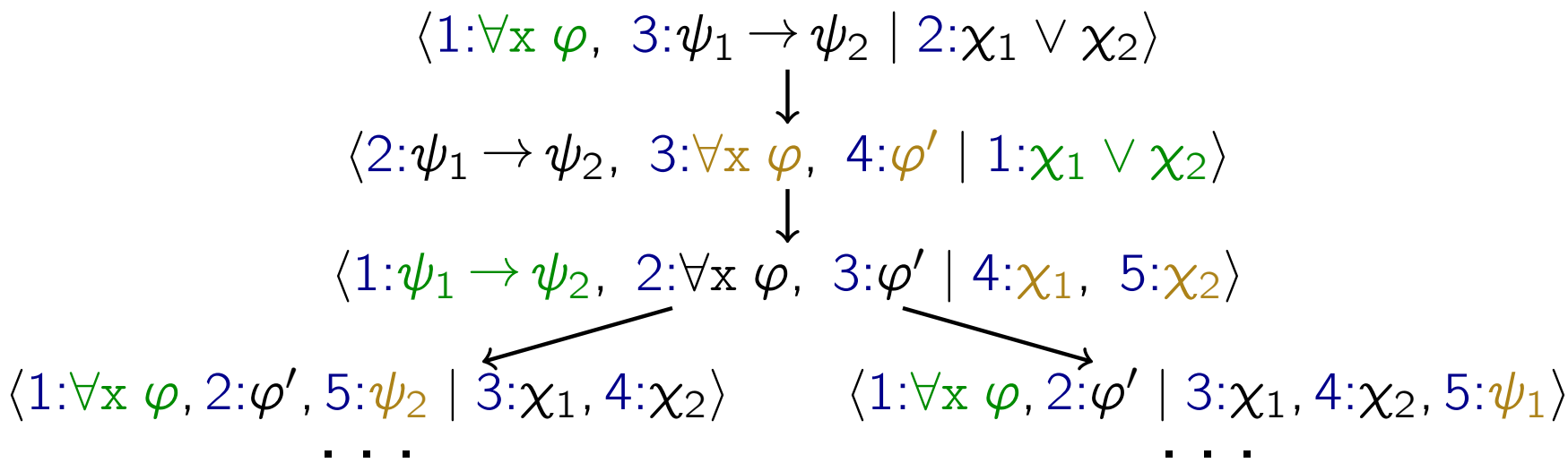
Доказательство теоремы о полноте табличного вывода.

Включим в стратегию построения успешного вывода три ограничения, разрешив выбирать остальные детали построения произвольно:

2. Правила применяются к формулам в порядке очереди

То есть:

- неатомарные формулы таблицы пронумерованы
- правило вывода применяется к первой формуле
- результат применения правила записывается последним



Тогда в каждой ветви вывода к каждой неатомарной формуле рано или поздно будет применено правило табличного вывода

Доказательство теоремы о полноте табличного вывода.

Включим в стратегию построения успешного вывода три ограничения, разрешив выбирать остальные детали построения произвольно:

3. При применении правил $L\forall$, $R\exists$ подставляются **все** имеющиеся в таблице константы (**c**, если констант нет)

$$\langle \forall \mathrm{x}\, \varphi,\ \exists \mathrm{x}\, \psi \mid \exists \mathrm{x}\, \chi \rangle$$
$$\downarrow L\forall$$
$$\langle \forall \mathrm{x}\, \varphi,\ \exists \mathrm{x}\, \psi,\ \varphi\{\mathrm{x}/\mathbf{c}\} \mid \exists \mathrm{x}\, \chi \rangle$$
$$\downarrow L\exists$$
$$\langle \forall \mathrm{x}\, \varphi,\ \psi\{\mathrm{x}/\mathbf{d}\},\ \varphi\{\mathrm{x}/\mathbf{c}\} \mid \exists \mathrm{x}\, \chi \rangle$$
$$\downarrow R\exists \times 2$$
$$\langle \forall \mathrm{x}\, \varphi,\ \psi\{\mathrm{x}/\mathbf{d}\},\ \varphi\{\mathrm{x}/\mathbf{c}\} \mid \exists \mathrm{x}\, \chi,\ \chi\{\mathrm{x}/\mathbf{c}\},\ \chi\{\mathrm{x}/\mathbf{d}\} \rangle$$
$$\ldots$$

Тогда в каждой бесконечной ветви вывода для каждого квантора $\forall$ слева и $\exists$ справа каждая константа рано или поздно будет подставлена

Доказательство теоремы о полноте табличного вывода.

Покажем, что любой вывод $\mathfrak{D}$, построенный для $T_0 = \langle \Gamma_0 \mid \Delta_0 \rangle$ согласно предложенной стратегии, успешен

Предположим, что это не так: вывод $\mathfrak{D}$ неуспешен — получим из этого выполнимость таблицы T_0, *противоречащую выбору таблицы* T_0

Заменим в $\mathfrak{D}$ каждую незакрытую атомарную таблицу T_{atom} на бесконечную ветвь $T_{atom} \to T_{atom} \to T_{atom} \to \ldots$

Тогда в полученном дереве обязательно найдётся бесконечная ветвь $\mathcal{T}$, состоящая только из незакрытых таблиц:

$$T_0 \to T_1 \to T_2 \to \ldots$$

По этой ветви построим интерпретацию $\mathcal{I}$, такую что:

- каждая формула из Γ_0 выполнима в $\mathcal{I}$
- каждая формула из Δ_0 невыполнима в $\mathcal{I}$

Доказательство теоремы о полноте табличного вывода.

$$\mathfrak{T}\colon\ T_0 \to T_1 \to T_2 \to \ldots, \qquad T_i = \langle \Gamma_i \mid \Delta_i \rangle$$

$\mathcal{I} = \langle \mathrm{D}, \overline{\mathrm{Const}}, -, \overline{\mathrm{Pred}} \rangle$, где

- предметная область — это все константы всех формул в $\mathfrak{T}$:
$$\mathrm{D} = \bigcup_{i \geq 0} \mathrm{Const}_i = \mathrm{Const}_\omega, \text{ где } \mathrm{Const}_i \text{ — все константы в } T_i$$
- значение каждой константы — это её изображение (*она сама*):
$$\overline{\mathbf{c}} = \mathbf{c}$$
- предикат истинен $\Leftrightarrow$ он встречается в левых частях таблиц $\mathfrak{T}$:
$$\overline{\mathrm{P}}(\mathbf{c_1}, \ldots, \mathbf{c_k}) = \mathbb{t} \qquad \Leftrightarrow \qquad \mathrm{P}(\mathbf{c_1}, \ldots, \mathbf{c_k}) \in \bigcup_{i \geq 0} \Gamma_i = \Gamma_\omega$$

Осталось показать *индукцией по структуре формулы*, что

- каждая формула из Γ_ω выполнима в $\mathcal{I}$
- каждая формула из $\Delta_\omega = \bigcup\limits_{i \geq 0} \Delta_i$ невыполнима в $\mathcal{I}$

Доказательство теоремы о полноте табличного вывода.

$$\mathfrak{T}\colon T_0 \to T_1 \to T_2 \to \ldots, \qquad T_i = \langle \Gamma_i \mid \Delta_i \rangle$$

$$\varphi \in \Gamma_\omega \stackrel{?}{\Rightarrow} \mathcal{I} \models \varphi \qquad\qquad \varphi \in \Delta_\omega \stackrel{?}{\Rightarrow} \mathcal{I} \not\models \varphi$$

База индукции: φ — атом

Тогда $\varphi = \mathrm{P}(\mathbf{c_1}, \ldots, \mathbf{c_k})$, где $\mathbf{c_1}, \ldots, \mathbf{c_k} \in \mathrm{Const}_\omega$ (почему?)

Подслучай 1: $\mathrm{P}(\mathbf{c_1}, \ldots, \mathbf{c_k}) \in \Gamma_\omega$
Тогда $\overline{\mathrm{P}}(\mathbf{c_1}, \ldots, \mathbf{c_k}) = \mathbb{t}$, а значит, $\mathcal{I} \models \varphi$

Подслучай 2: $\mathrm{P}(\mathbf{c_1}, \ldots, \mathbf{c_k}) \in \Delta_\omega$
Тогда $\mathrm{P}(\mathbf{c_1}, \ldots, \mathbf{c_k}) \notin \Gamma_\omega$ (почему?)
Значит, $\overline{\mathrm{P}}(\mathbf{c_1}, \ldots, \mathbf{c_k}) = \mathbb{f}$ и $\mathcal{I} \not\models \varphi$

Индуктивный переход
Предположение индукции: для каждой формулы, содержащей менее N логических операций, утверждение доказано
Рассматриваемый случай: формула φ содержит ровно N логических операций

Доказательство теоремы о полноте табличного вывода.

$$\mathfrak{T}\colon T_0 \to T_1 \to T_2 \to \ldots, \qquad T_i = \langle \Gamma_i \mid \Delta_i \rangle$$

$$\varphi \in \Gamma_\omega \overset{?}{\Rightarrow} \mathcal{I} \models \varphi \qquad\qquad \varphi \in \Delta_\omega \overset{?}{\Rightarrow} \mathcal{I} \not\models \varphi$$

Переход 1: $\varphi = \psi \to \chi$

Подслучай 1: $\varphi \in \Gamma_\omega$

В ветви $\mathfrak{T}$ существует таблица T_i, такая что правило вывода применяется к φ в левой части этой таблицы (почему?)

Значит, верно хотя бы одно из двух: (почему?)

- $\chi \in \Gamma_{i+1}$, и тогда $\mathcal{I} \models \chi$ и $\mathcal{I} \models \varphi$
- $\psi \in \Delta_{i+1}$, и тогда $\mathcal{I} \not\models \psi$ и $\mathcal{I} \models \varphi$

Подслучай 2: $\varphi \in \Delta_\omega$ — рассуждения аналогичны

Переход 2/3/4: $\varphi = \psi \,\&\, \chi$ / $\psi \vee \chi$ / $\neg\psi$ — рассуждения аналогичны

Доказательство теоремы о полноте табличного вывода.

$$\mathfrak{T}\colon T_0 \to T_1 \to T_2 \to \ldots, \qquad T_i = \langle \Gamma_i \mid \Delta_i \rangle$$

$$\varphi \in \Gamma_\omega \overset{?}{\Rightarrow} \mathcal{I} \models \varphi \qquad\qquad \varphi \in \Delta_\omega \overset{?}{\Rightarrow} \mathcal{I} \not\models \varphi$$

Переход 5: $\varphi = \forall \mathrm{x}\, \psi$

Подслучай 1: $\varphi \in \Gamma_\omega$

Тогда $\psi\{\mathrm{x}/\mathbf{c}\} \in \Gamma_\omega$ для любой константы $\mathbf{c} \in \mathrm{Const}_\omega$ (почему?)

Значит, для любой константы $\mathbf{c} \in \mathrm{Const}_\omega$ верно: $\mathcal{I} \models \psi\{\mathrm{x}/\mathbf{c}\}$

Но это и означает $\mathcal{I} \models \forall \mathrm{x}\, \psi$

Подслучай 2: $\varphi \in \Delta_\omega$

В ветви $\mathfrak{T}$ существует таблица T_i, такая что правило вывода применяется к φ в правой части этой таблицы

Значит, $\psi\{\mathrm{x}/\mathbf{c}\} \in \Delta_{i+1}$ для некоторой $\mathbf{c} \in \mathrm{Const}_{i+1} \subseteq \mathrm{Const}_\omega$

Тогда $\mathcal{I} \not\models \psi\{\mathrm{x}/\mathbf{c}\}$, и следовательно, $\mathcal{I} \not\models \forall \mathrm{x}\, \psi$

Переход 6: $\varphi = \exists \mathrm{x}\, \psi$ — рассуждения аналогичны

Доказательство теоремы о полноте табличного вывода.

$$\mathfrak{T}\colon T_0 \to T_1 \to T_2 \to \ldots, \qquad T_i = \langle \Gamma_i \mid \Delta_i \rangle$$

Итоги рассуждений

Существует (*и явно описана*) интерпретация $\mathcal{I}$, такая что

- все формулы в левых частях таблиц из $\mathfrak{T}$ выполнимы в $\mathcal{I}$
- все формулы в правых частях таблиц из $\mathfrak{T}$ невыполнимы в $\mathcal{I}$

В частности, все формулы из Γ_0 выполнимы в $\mathcal{I}$
и все формулы из Δ_0 невыполнимы в $\mathcal{I}$

Значит, таблица T_0 выполнима, *что противоречит невыполнимости этой таблицы, заявленной в начале доказательства*

Противоречие получено в предположении о том,
что вывод, построенный для T_0, неуспешен

Значит, предположение неверно:
любой вывод, построенный для невыполнимой таблицы T_0 согласно предложенным правилам, успешен ▼

Доказательство теоремы о полноте табличного вывода.

Интересующимся для самостоятельного размышления:
а как адаптировать доказательство к общему случаю?

То есть:

- Какой порядок обработки формул позволит «справедливо» обращаться со счётными множествами формул?
- Какие термы подставлять, если в сигнатуре алфавита есть функциональные символы?
- Как задать и использовать интерпретацию $\mathcal{I}$, если в таблицах встречаются незамкнутые формулы?

Следствие

Семантическая таблица T логики предикатов невыполнима $\Leftrightarrow$ для неё существует успешный табличный вывод

Доказательство.

Вытекает из теорем о корректности и полноте табличного вывода ▼

Следствие. **Для любой формулы φ логики предикатов верно:**

$$\models \varphi \quad \Leftrightarrow \quad \text{для семантической таблицы } \langle \mid \varphi \rangle \text{ существует успешный табличный вывод}$$

Доказательство. Вытекает из первого следствия и теоремы о табличной проверке общезначимости формул ▼

Математическая логика и логическое программирование

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

Блок 13

Теорема Лёвенгейма-Сколема
Теорема компактности Мальцева
Автоматизация доказательства теорем

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Корректность и **полнота** метода семантических таблиц в ЛП:
$\models \varphi \Leftrightarrow$ **для таблицы** $\langle \mid \varphi \rangle$ **существует успешный табличный вывод**

Полнота метода семантических таблиц в ЛП (доказательство):
если $\models \varphi$**, то успешный табличный вывод можно построить, придерживаясь особой стратегии**

Свойства и «приёмы», обсуждавшиеся
для метода семантических таблиц, позволяют

- обосновать несколько нетривиальных утверждений,
  не имеющих прямого отношения к методу, и
- естественно поставить важные алгоритмические вопросы,
  касающиеся проблемы общезначимости формул логики предикатов

Теорема Лёвенгейма-Сколема

Вспомним определение выполнимости формул логики предикатов:
Формула выполнима $\Leftrightarrow$ она выполняется хотя бы в одной интерпретации

Представим себе, что для проверки выполнимости формулы
разрешено перебрать **сколько угодно** интерпретаций
и в каждой проверить выполнимость формулы

Следует ли перебрать все существующие в природе интерпретации,
или же достаточно ограничиться только какими-нибудь «простыми»?

В таком переборе намного важнее природы предметов
оказывается их **количество** (как было отмечено, например, в блоке 7)

Хотелось бы в полноценных рассуждениях о выполнимости формулы
использовать только интерпретации с хотя и бесконечной,
но всё же как можно меньшей предметной областью

Теорема Лёвенгейма-Сколема

Для любого предложения φ справедлива равносильность:

φ выполнимо $\Leftrightarrow$ φ имеет модель с не более чем счётной предметной областью

Доказательство.

Совместим знания о корректности и полноте вывода
и определение выполнимости таблицы:
$\not\models \varphi \Leftrightarrow$ для $T_0 = \langle \varphi \mid \rangle$ не существует успешного табличного вывода

Построив вывод согласно стратегии
из доказательства теоремы о полноте, получим

- бесконечную ветвь $T_0 \to T_1 \to \ldots$ вывода,
  состоящую только из незакрытых таблиц
- интерпретацию $\mathcal{I}$ с **не более чем счётной** предметной областью,
  в которой выполняются все таблицы этой ветви —
  в том числе и таблица T_0 ▼

Теорема компактности Мальцева

Вспомним теорему о логическом следствии:
методы проверки общезначимости формул можно применить
и для проверки логического следования одних формул из других

$$\psi_1, \ldots, \psi_k \models \varphi \Leftrightarrow \models \psi_1 \,\&\, \ldots \,\&\, \psi_k \to \varphi$$

Здесь $\{\psi_1, \ldots, \psi_k\}$ — это **конечная** база знаний, относительно которой требуется проверить достоверность извлечённого следствия

А можно ли предложить что-нибудь аналогичное
для **бесконечных** баз знаний?

Оказывается, что, независимо от размера набора знаний,
в достоверности логического следствия можно убедиться,
выбрав для рассмотрения только некоторый **конечный** поднабор

Теорема компактности Мальцева

Для любого предложения φ и любого множества предложений Γ справедлива равносильность:

$\Gamma \models \varphi \quad \Leftrightarrow$ **существует конечное подмножество Γ' множества Γ, такое что $\Gamma' \models \varphi$**

Доказательство.

$\Gamma \models \varphi \Leftrightarrow$ таблица $T = \langle \Gamma \mid \varphi \rangle$ невыполнима (почему?)
$\Leftrightarrow$ существует успешный табличный вывод $\mathfrak{D}$ для T

Подмножество Γ_1 всех формул множества Γ, к которым применяются правила вывода в $\mathfrak{D}$, конечно, как и минимальное подмножество Γ_2, такое что для каждого листа $\mathfrak{D}$ существует формула из Γ_2, содержащаяся в обеих частях таблицы этого листа (почему?)

Тогда для таблицы $\langle \Gamma_1 \cup \Gamma_2 \mid \varphi \rangle$,
также существует успешный табличный вывод (почему?)

Значит, $\Gamma_1 \cup \Gamma_2 \models \varphi$ ▼

Теорема компактности Мальцева

Множество предложений Γ называется выполнимым ($\models\!\!|\ \Gamma$[1]), если у этого множества есть хотя бы одна модель

Следствие. Для любого множества предложений Γ верно:

$$\not\models \Gamma$$

$$\Leftrightarrow$$

существует конечное подмножество Γ' множества Γ, такое что $\not\models \Gamma'$

Доказательство.

$\not\models \Gamma$

$\Leftrightarrow$ (определения логического следования и невыполнимости)

$\Gamma \models \varphi$ для произвольно взятого невыполнимого предложения φ

$\Leftrightarrow$ (теорема компатности Мальцева)

существует подмножество Γ' множества Γ, такое что $\Gamma' \models \varphi$ (для произвольно взятого невыполнимого предложения φ)

$\Leftrightarrow$ (определения логического следования и невыполнимости)

существует подмножество Γ' множества Γ, такое что $\not\models \Gamma'$ ▼

1 Снова то же необщеупотребимое обозначение

Автоматизация доказательства теорем

В свете того, что есть стратегия построения успешного табличного вывода для произвольной невыполнимой таблицы, естественно возникает вопрос:

А можно ли поручить проверку общезначимости формул ЛП компьютеру, чтобы он делал всю работу за нас?

Если формула общезначима, то это можно обосновать, придерживаясь упомянутой стратегии

А если необщезначима ... (?)
... то на этот счёт пока есть только теорема о корректности:
успешного вывода для соответствующей таблицы не существует

Познакомившись получше с логикой предикатов и логическими программами, обсудим и то, что целиком переложить такую работу на компьютер **невозможно**[1]

Но если всё же попытаться, то ...

1 То есть о том, что проблема общезначимости формул *алгоритмически неразрешима*

Автоматизация доказательства теорем

Если программно реализовать стратегию построения логического вывода[1], то в результате получится прувер: средство доказательства теорем логики предикатов

First-order theorem prover

К пруверу разумно было бы предъявить такие требования:

- корректность: выдаются только правильные ответы — обязательно
- полнота: ответы выдаются всегда — желательно как можно лучше к этому приблизиться
- эффективность: ответы выдаются за разумное время — очень желательно

1 Не обязательно стратегию из доказательства теоремы полноты. Не обязательно полную стратегию. Не обязательно табличного вывода

Автоматизация доказательства теорем

Если программно реализовать стратегию построения логического вывода[1], то в результате получится прувер: средство доказательства теорем логики предикатов

First-order theorem prover

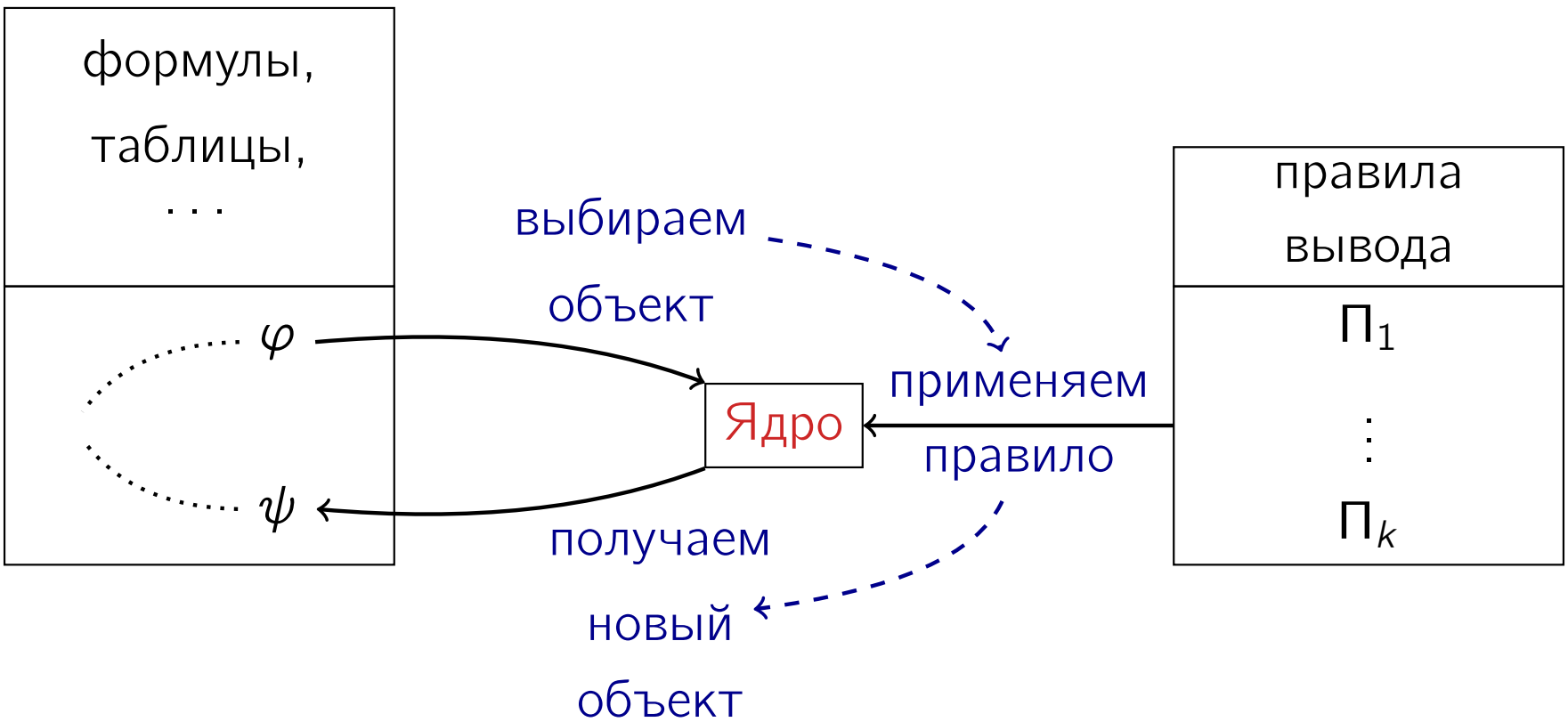
Один из **очень многих** примеров того, чего позволило добиться использование пруверов:[2] строго доказана корректность *nix-микроядра L4, и в процессе доказательства найдены и исправлены сотни ошибок в коде

1 Не обязательно стратегию из доказательства теоремы полноты. Не обязательно полную стратегию. Не обязательно табличного вывода

2 Klein et al. seL4: formal verification of an OS kernel. 2009. Конкретно этот пример выбран только из-за наглядности, понятности и при этом «неоспоримой полезности» формулировки результата

Автоматизация доказательства теорем

Как устроены пруверы:



Автоматизация доказательства теорем

Представим себе прувер, способный проверять общезначимость формул логики предикатов **методом семантических таблиц**:

- корректный
- выдающий ответ «да» для **всех** общезначимых формул

Насколько эффективен может быть такой прувер?

Эффективность построения вывода определяется тем,

- как на каждом шаге выбираются формулы для применения к ним правил и
- какие термы подставляются при применении правил $L\forall$ и $R\exists$

Если прувером осуществляется полный перебор всех формул, возникающих в таблицах, или перебор слишком большого числа термов, то такой прувер, вероятно, окажется неэффективным

Автоматизация доказательства теорем

Избежать полного перебора формул — непростая задача:

База знаний	Запрос
В огороде бузина	В Киеве дядька
Растёт(**бузина**, **огород**)	$\exists$у (Дядька(у) & Живёт(у, **Киев**))

Автоматизация доказательства теорем

Избежать полного перебора формул — непростая задача:

База знаний	Запрос
В огороде бузина Растёт(**бузина**, **огород**) Всё в огороде посадил дядька $\forall$x ($\text{Растёт}(\text{x}, \textbf{огород}) \rightarrow$ $\exists$y ($\text{Посадил}(\text{y}, \text{x}) \,\&\, \text{Дядька}(\text{y})$))	В Киеве дядька $\exists$y ($\text{Дядька}(\text{y})$ $\,\&\, \text{Живёт}(\text{y}, \textbf{Киев})$)

Автоматизация доказательства теорем

Избежать полного перебора формул — непростая задача:

База знаний	Запрос
В огороде бузина Растёт(**бузина**, **огород**)	В Киеве дядька $\exists y\ (\text{Дядька}(y)$ $\quad \& \text{Живёт}(y, \textbf{Киев}))$
Всё в огороде посадил дядька $\forall x\ (\text{Растёт}(x, \textbf{огород}) \rightarrow$ $\quad \exists y\ (\text{Посадил}(y, x) \& \text{Дядька}(y)))$	
Бузину сажают только Киевляне $\forall x\ (\text{Посадил}(x, \textbf{бузина})$ $\quad \rightarrow \text{Живёт}(x, \textbf{Киев}))$	

Автоматизация доказательства теорем

$$L\forall\ \frac{\langle \Gamma,\ \forall x\ \varphi(x) \mid \Delta\rangle}{\langle \Gamma,\ \forall x\ \varphi(x),\ \varphi(x)\{x/t\} \mid \Delta\rangle} \qquad R\exists\ \frac{\langle \Gamma \mid \Delta,\ \exists x\ \varphi(x)\rangle}{\langle \Gamma \mid \Delta,\ \exists x\ \varphi(x),\ \varphi(x)\{x/t\}\rangle}$$

Избежать перебора большого числа термов — непростая задача

Для примера представим себе, что при построении вывода придётся перебрать все термы, составленные из **одного** функционального символа $\mathbf{f}^{(2)}$, используемого не более 10 раз, и **двух** различных констант (*вроде бы это не очень большие термы?*)

Можно легко посчитать, что существует **более** 10^{300} различных термов такого вида

Число 10^{100} (на 200 нолей меньше) имеет особое название — гугол: это бессмысленно большое число, превосходящее число атомов в наблюдаемой вселенной

Автоматизация доказательства теорем

$$L\forall\ \frac{\langle \Gamma,\ \forall x\ \varphi(x) \mid \Delta\rangle}{\langle \Gamma,\ \forall x\ \varphi(x),\ \varphi(x)\{x/t\} \mid \Delta\rangle} \qquad R\exists\ \frac{\langle \Gamma \mid \Delta,\ \exists x\ \varphi(x)\rangle}{\langle \Gamma \mid \Delta,\ \exists x\ \varphi(x),\ \varphi(x)\{x/t\}\rangle}$$

Избежать перебора большого числа термов — непростая задача

Существуют способы повышения эффективности
перебора термов при построении логического вывода
для доказательства общезначимости формул[1]

Далее обсудим один из таких способов и основанный на нём метод:
метод резолюций

Заодно в процессе обсуждения познакомимся
и с другими важными понятиями и задачами и методами,
касающимися логики предикатов, и подготовим основу
для обсуждения логического программирования

1 J.A. Robinson: метод резолюций. С.Ю. Маслов: обратный вывод

Математическая логика и логическое программирование

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

Блок 14

Общая схема метода резолюций

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Проблему общезначимости формул логики предикатов

$$\models \varphi?$$

можно *до некоторой степени* решить с помощью
метода семантических таблиц

Но этот метод оказался **неэффективным**:
в «плохих» случаях может потребоваться

- перебирать много формул и
- подставлять много термов

На ближайших лекциях будет обсуждаться более эффективный метод проверки общезначимости формул логики предикатов:

метод резолюций

Без ограничения общности далее полагаем, что формула φ, общезначимость которой проверяется методом резолюций, замкнута

Общая схема метода резолюций

Сквозной пример: обоснование общезначимости формулы

$$\models \exists \mathrm{x}\,(\mathrm{P}(\mathrm{x}) \,\&\, (\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$

Этап 1: перейти к проверке невыполнимости отрицания формулы

$$\cdots \rightsquigarrow \neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x}) \,\&\, (\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$

Этап 2: упростить формулу, сохранив её смысл (привести к ПНФ)

$$\cdots \rightsquigarrow \forall \mathrm{x}\,\exists \mathrm{z}\,\exists \mathrm{y}\,\forall \mathrm{u}\,(\mathrm{P}(\mathrm{x}) \,\&\, (\neg \mathrm{P}(\mathrm{z}) \vee \mathrm{R}(\mathrm{x},\mathrm{y})) \,\&\, \neg \mathrm{R}(\mathrm{x},\mathrm{u}))$$

Этап 3: сделать формулу ещё проще *с изменением смысла*, сохранив её выполнимость/невыполнимость (привести к ССФ)

$$\cdots \rightsquigarrow \forall \mathrm{x}\,\forall \mathrm{u}\,(\mathrm{P}(\mathrm{x}) \,\&\, (\neg \mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x},\mathbf{g}(\mathrm{x}))) \,\&\, \neg \mathrm{R}(\mathrm{x},\mathrm{u}))$$

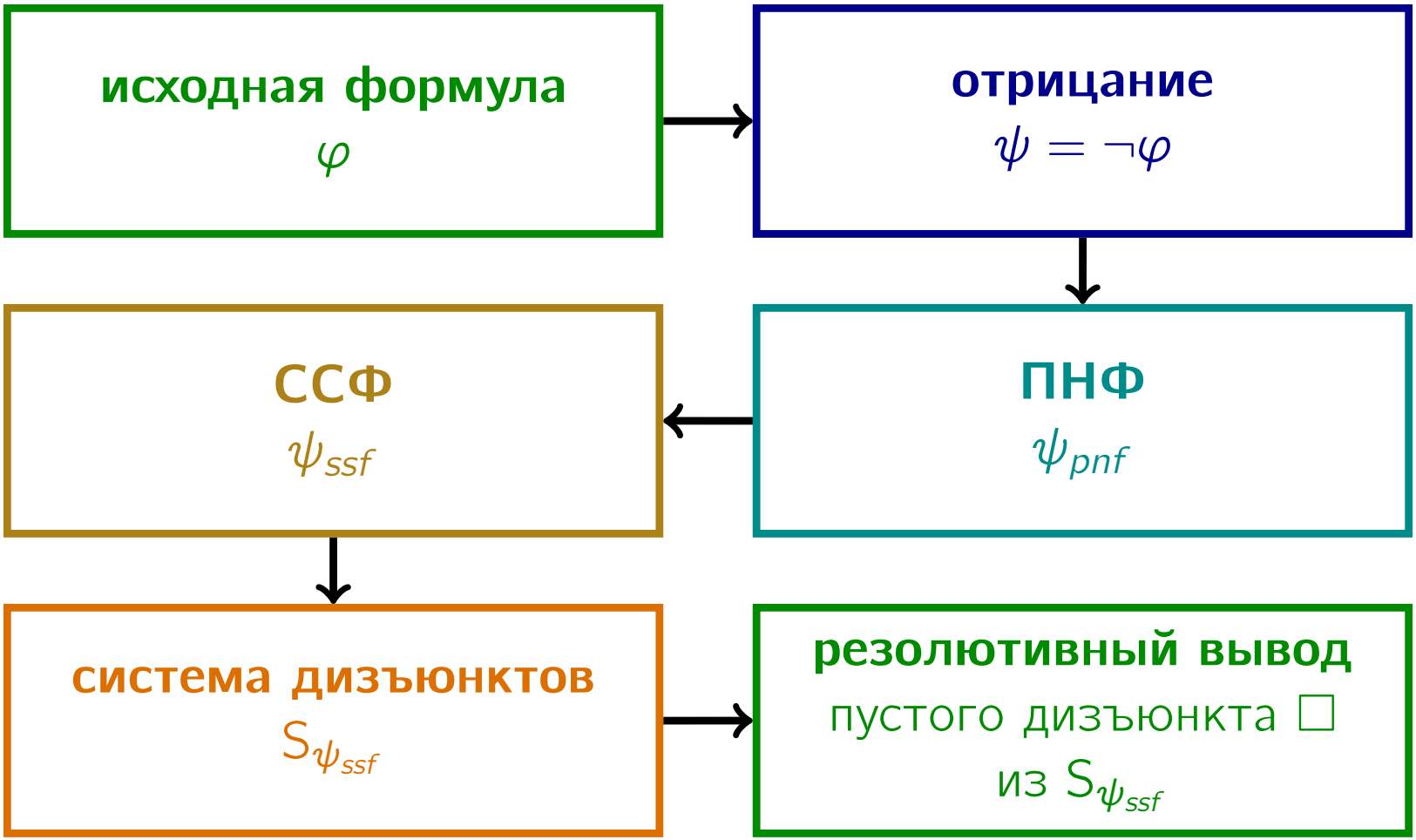
Этап 4: перейти к проверке невыполнимости системы *очень* простых формул — дизъюнктов

$$\cdots \rightsquigarrow \{\mathrm{P}(\mathrm{x}),\ \neg \mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x},\mathbf{g}(\mathrm{x})),\ \neg \mathrm{R}(\mathrm{x},\mathrm{u})\}$$

Этап 5: построить логический вывод особого дизъюнкта ($\square$), обозначающего невыполнимость системы

$$\neg \mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x},\mathbf{g}(\mathrm{x})) \quad \underbrace{\mathrm{P}(\mathrm{x}') \to \mathrm{R}(\mathrm{x},\mathbf{g}(\mathrm{x}))}_{\{\mathrm{x}'/\mathbf{f}(\mathrm{x})\}} \quad \underbrace{\neg \mathrm{R}(\mathrm{x}',\mathrm{u}') \to \square}_{\{\mathrm{x}'/\mathrm{x},\,\mathrm{u}'/\mathbf{f}(\mathrm{x})\}}$$

Общая схема метода резолюций



Отдельно будет показана справедливость цепочки равносильностей:

$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf} \quad \Leftrightarrow \quad \not\models S_{\psi_{ssf}}$$

$$\Leftrightarrow \quad \text{существует вывод } \square \text{ из } S_{\psi_{ssf}}$$

Математическая логика и логическое программирование

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

Блок 15

Равносильность формул

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Из булевой алгебры:

$$A \vee B \to (C \,\&\, D \to E)$$
$$\mapsto \qquad (x \to y \mapsto \neg x \vee y)$$
$$\neg(A \vee B) \vee \neg(C \,\&\, D) \vee E$$
$$\mapsto \qquad (\neg(x \,\&\, y) \mapsto \neg x \vee \neg y;\ \ldots \mapsto \ldots)$$
$$\neg A \,\&\, \neg B \vee \neg C \vee \neg D \vee E$$
$$\mapsto \qquad (\ldots \mapsto \ldots)$$
$$(\neg A \vee \neg C \vee \neg D \vee E) \,\&\, (\neg B \vee \neg C \vee \neg D \vee E)$$

Нижняя формула получена из верхней
при помощи основных тождеств булевой алгебры

Значит, можно быть уверенным в том,
что эти формулы имеют одинаковый смысл

Неплохо было бы (*и для метода резолюций, и в целом*)
уметь преобразовывать формулы **логики предикатов**
с гарантированным сохранением их смысла

Равносильность формул

Эквивалентность (логическая связка):

$\varphi \leftrightarrow \psi$ — это сокращение для формулы $(\varphi \to \psi)\,\&\,(\psi \to \varphi)$

Формулы φ, ψ равносильны ($\varphi \sim \psi$), если формула $\varphi \leftrightarrow \psi$ общезначима

Утверждение. Для любых равносильных формул $\varphi(\widetilde{\mathrm{x}^{\mathrm{n}}})$, $\psi(\widetilde{\mathrm{x}^{\mathrm{n}}})$ ЛП, интерпретации $\mathcal{I}$ и набора предметов $\widetilde{d^n}$ верно следующее:

$$\mathcal{I} \models \varphi[\widetilde{d^n}] \quad \Leftrightarrow \quad \mathcal{I} \models \psi[\widetilde{d^n}]$$

Доказательство.

($\Rightarrow$) Пусть $\mathcal{I} \models \varphi[\widetilde{d^n}]$. Тогда:

$\models (\varphi \to \psi)\,\&\,(\psi \to \varphi)$ (по определению равносильности)

$\mathcal{I} \models ((\varphi \to \psi)\,\&\,(\psi \to \varphi))[\widetilde{d^n}]$ (по определению общезначимости)

$\mathcal{I} \models (\varphi \to \psi)[\widetilde{d^n}]$ (по семантике «&»)

$\mathcal{I} \models \psi[\widetilde{d^n}]$ (по семантике «$\to$» и соотношению $\mathcal{I} \models \varphi[\widetilde{d^n}]$)

($\Leftarrow$) Аналогично ▼

Равносильность формул

Утверждение. $\sim$ **— отношение эквивалентности**

Утверждение. **Если формула** φ **общезначима, то любая равносильная ей формула** ψ **также общезначима**

Утверждение. **Если формула** φ **выполнима, то любая равносильная ей формула** ψ **также выполнима**

Доказательства опустим для экономии времени

Основные равносильности

Законы булевой алгебры

Коммутативность $\&$ и $\vee$:

$$\varphi \,\&\, \psi \sim \psi \,\&\, \varphi \qquad \varphi \vee \psi \sim \psi \vee \varphi$$

Ассоциативность $\&$ и $\vee$:

$$(\varphi \,\&\, \psi) \,\&\, \chi \sim \varphi \,\&\, (\psi \,\&\, \chi) \qquad (\varphi \vee \psi) \vee \chi \sim \varphi \vee (\psi \vee \chi)$$

Дистрибутивность $\vee$ относительно $\&$ и $\&$ относительно $\vee$:

$$\varphi \,\&\, (\psi \vee \chi) \sim \varphi \,\&\, \psi \vee \varphi \,\&\, \chi \qquad \varphi \vee (\psi \,\&\, \chi) \sim (\varphi \vee \psi) \,\&\, (\varphi \vee \chi)$$

Идемпотентность $\&$ и $\vee$:

$$\varphi \,\&\, \varphi \sim \varphi \qquad \varphi \vee \varphi \sim \varphi$$

Инволютивность $\neg$:

$$\neg\neg\varphi \sim \varphi$$

Законы де Моргана для $\&$ и для $\vee$:

$$\neg(\varphi \,\&\, \psi) \sim \neg\varphi \vee \neg\psi \qquad \neg(\varphi \vee \psi) \sim \neg\varphi \,\&\, \neg\psi$$

Закон удаления импликации:

$$\varphi \to \psi \sim \neg\varphi \vee \psi$$

... ...

Основные равносильности

Правила работы с кванторами

Переименование связанной переменной:

$$\forall \mathrm{x}\ \varphi \sim \forall \mathrm{y}\ (\varphi\{\mathrm{x}/\mathrm{y}\}) \qquad \exists \mathrm{x}\ \varphi \sim \exists \mathrm{y}\ (\varphi\{\mathrm{x}/\mathrm{y}\})$$

(*если* $\mathrm{y} \notin \mathrm{Var}_\varphi$ *и подстановка* $\{\mathrm{x}/\mathrm{y}\}$ *правильна для* φ)

Продвижение отрицания под квантор:

$$\neg\forall \mathrm{x}\ \varphi \sim \exists \mathrm{x}\ \neg\varphi \qquad \neg\exists \mathrm{x}\ \varphi \sim \forall \mathrm{x}\ \neg\varphi$$

Вынесение квантора за скобки:

$$\forall \mathrm{x}\ \varphi \,\&\, \psi \sim \forall \mathrm{x}\ (\varphi \,\&\, \psi) \qquad \exists \mathrm{x}\ \varphi \,\&\, \psi \sim \exists \mathrm{x}\ (\varphi \,\&\, \psi)$$

$$\forall \mathrm{x}\ \varphi \vee \psi \sim \forall \mathrm{x}\ (\varphi \vee \psi) \qquad \exists \mathrm{x}\ \varphi \vee \psi \sim \exists \mathrm{x}\ (\varphi \vee \psi)$$

(*если* $\mathrm{x} \notin \mathrm{Var}_\psi$)

Строго говоря, справедливость каждой упомянутой равносильности следует **обосновать**

Но эти обоснования устроены очень просто и однотипно, и потому здесь не приводятся: достаточно применить метод семантических таблиц

Теорема о равносильной замене

$\varphi[\![\psi]\!]$ — обозначение формулы φ, содержащей подформулу ψ

$\varphi[\![\psi/\chi]\!]$ — формула, получающаяся из φ заменой некоторого вхождения подформулы ψ на χ

Теорема (о равносильной замене в ЛП)
Для любых формул φ, ψ, χ логики предикатов верно:

$$\psi \sim \chi \quad \Rightarrow \quad \varphi[\![\psi]\!] \sim \varphi[\![\psi/\chi]\!]$$

Доказательство (индукцией по структуре формулы φ).

База индукции: $\varphi = \psi$ — очевидно $(\psi \sim \chi \Rightarrow \varphi = \psi \sim \chi)$

Индуктивный переход. Подробно разберём только один случай:

$$\varphi(\widetilde{\mathrm{x}^{\mathrm{n}}}) = \forall \mathrm{x}\, (\varphi'[\![\psi]\!])$$

Остальные случаи аналогичны

Теорема о равносильной замене

Доказательство (индуктивный переход).

Утверждение: $\forall \mathrm{x}\ (\varphi'[\![\psi]\!]) \sim \forall \mathrm{x}\ (\varphi'[\![\psi/\chi]\!])$

Индуктивное предположение: $\varphi'[\![\psi]\!] \sim \varphi'[\![\psi/\chi]\!]$, то есть
для любой интерпретации $\mathcal{I}$ и любых предметов d, $\widetilde{d}^n$ верно:

$$\mathcal{I} \models (\varphi'[\![\psi]\!] \to \varphi'[\![\psi/\chi]\!])[\mathrm{x}/d, \widetilde{\mathrm{x}}^{\mathrm{n}}/\widetilde{d}^n]$$
$$\mathcal{I} \models (\varphi'[\![\psi/\chi]\!] \to \varphi'[\![\psi]\!])[\mathrm{x}/d, \widetilde{\mathrm{x}}^{\mathrm{n}}/\widetilde{d}^n]$$

Тогда

$$\mathcal{I} \models \forall \mathrm{x}\ (\varphi'[\![\psi]\!] \to \varphi'[\![\psi/\chi]\!])[\widetilde{\mathrm{x}}^{\mathrm{n}}/\widetilde{d}^n]$$
$$\mathcal{I} \models \forall \mathrm{x}\ (\varphi'[\![\psi/\chi]\!] \to \varphi'[\![\psi]\!])[\widetilde{\mathrm{x}}^{\mathrm{n}}/\widetilde{d}^n]$$

Из блока 10: $\models \forall \mathrm{x}\ (A \to B) \to (\forall \mathrm{x}\ A \to \forall \mathrm{x}\ B)$, а значит:

$$\mathcal{I} \models (\forall \mathrm{x}\ \varphi'[\![\psi]\!] \to \forall \mathrm{x}\ \varphi'[\![\psi/\chi]\!])[\widetilde{\mathrm{x}}^{\mathrm{n}}/\widetilde{d}^n]$$
$$\mathcal{I} \models (\forall \mathrm{x}\ \varphi'[\![\psi/\chi]\!] \to \forall \mathrm{x}\ \varphi'[\![\psi]\!])[\widetilde{\mathrm{x}}^{\mathrm{n}}/\widetilde{d}^n]$$

Но это и есть $\forall \mathrm{x}\ (\varphi'[\![\psi]\!]) \sim \forall \mathrm{x}\ (\varphi'[\![\psi/\chi]\!])$ ▼

Пример напоследок

Используя равносильную замену, можно существенно изменить форму высказывания, сохранив его смысл — (не)выполнимость в каждой интерпретации на каждом наборе предметов

Например,

$$\forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \to \exists \mathrm{x}\, \mathrm{P}(\mathrm{x})$$

$$\sim \qquad (\exists \mathrm{x}\, \varphi \sim \exists \mathrm{y}\, (\varphi\{\mathrm{x}/\mathrm{y}\}))$$

$$\forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\, \mathrm{P}(\mathrm{y})$$

$$\sim \qquad (\varphi \to \psi \sim \neg\varphi \vee \psi)$$

$$\neg\forall \mathrm{x}\, \mathrm{P}(\mathrm{x}) \vee \exists \mathrm{y}\, \mathrm{P}(\mathrm{y})$$

$$\sim \qquad (\neg\forall \mathrm{x}\, \varphi \sim \exists \mathrm{x}\, \neg\varphi)$$

$$\exists \mathrm{x}\, \neg\mathrm{P}(\mathrm{x}) \vee \exists \mathrm{y}\, \mathrm{P}(\mathrm{y})$$

$$\sim \qquad (\exists \mathrm{x}\, \varphi \vee \psi \sim \exists \mathrm{x}\, (\varphi \vee \psi);\ \varphi \vee \psi \sim \psi \vee \varphi)$$

$$\exists \mathrm{x}\, \exists \mathrm{y}\, (\neg\mathrm{P}(\mathrm{x}) \vee \mathrm{P}(\mathrm{y}))$$

$$\sim \qquad (\varphi \to \psi \sim \neg\varphi \vee \psi)$$

$$\exists \mathrm{x}\, \exists \mathrm{y}\, (\mathrm{P}(\mathrm{x}) \to \mathrm{P}(\mathrm{y}))$$

Математическая логика и логическое программирование

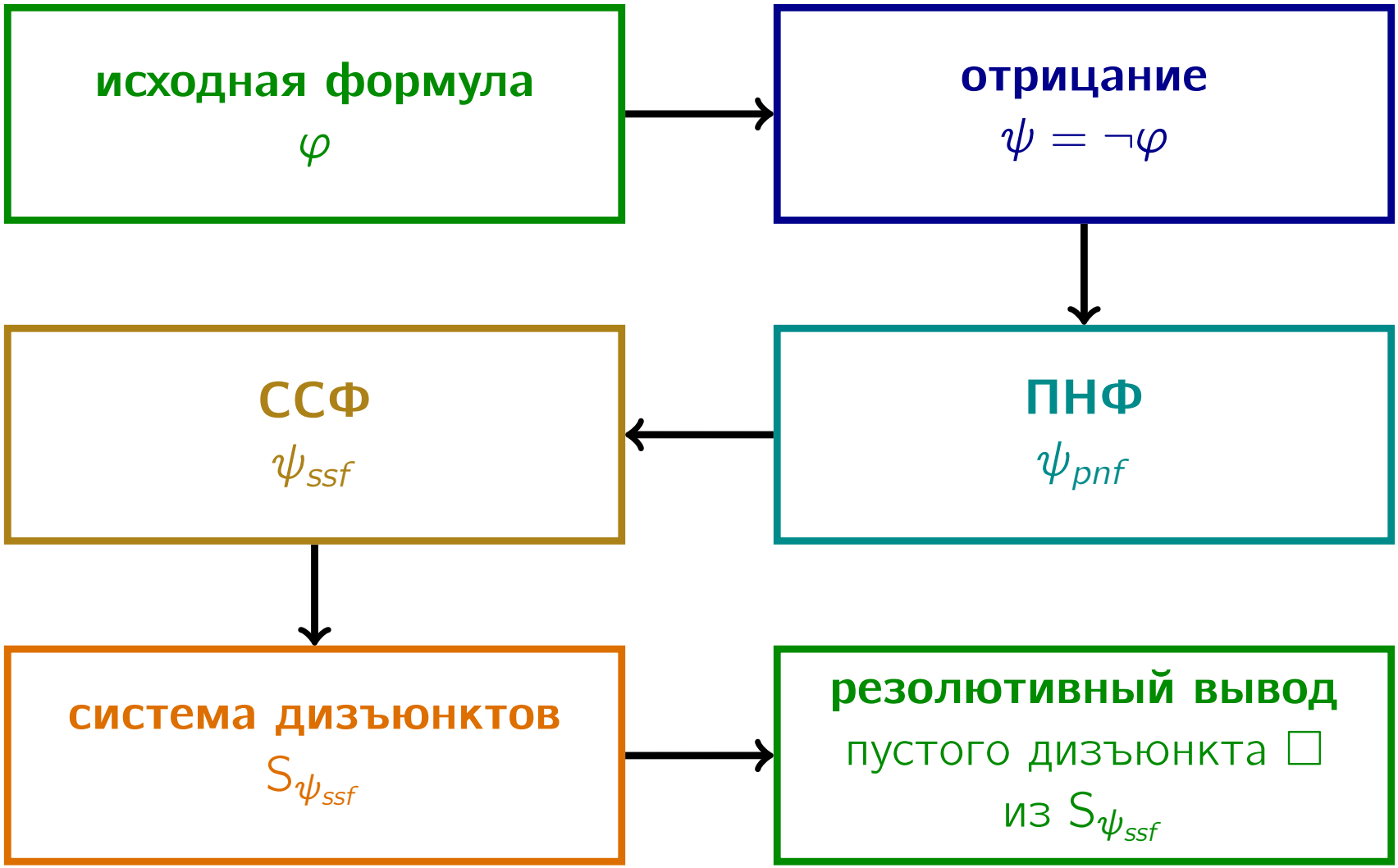
mk.cs.msu.ru → Лекционные курсы
→ Математическая логика и логическое программирование (3-й поток)

Блок 16

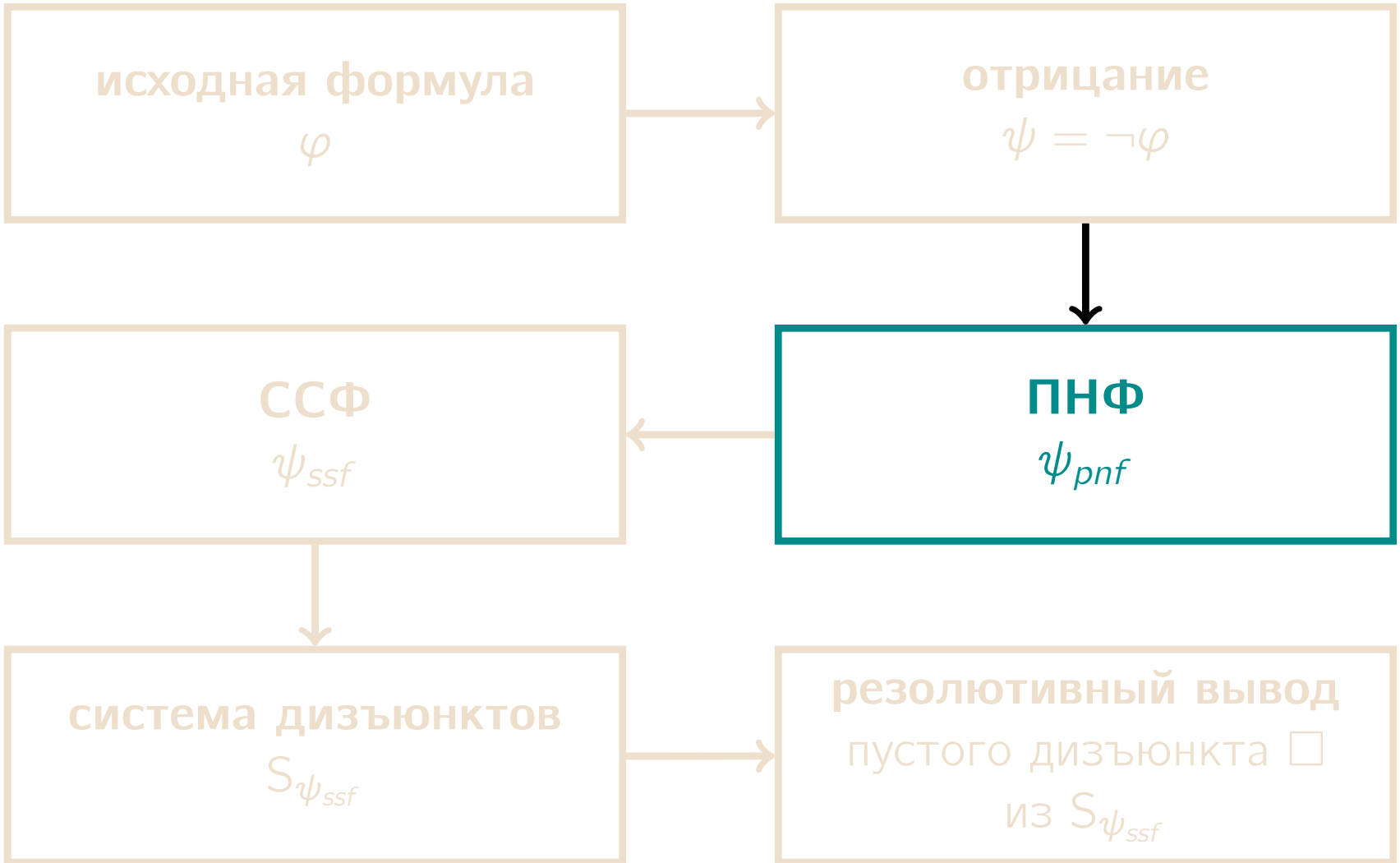
Предварённая нормальная форма (ПНФ)

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

ВМК МГУ, 2023/2024, осенний семестр



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi$$



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi$$

Предварённая нормальная форма

Замкнутая формула логики предикатов находится в предварённой нормальной форме (ПНФ), если она имеет вид

$$\underbrace{Q_1 \mathtt{x}_1 \ \ldots \ Q_n \mathtt{x}_\mathtt{n}}_{\text{кванторная приставка}} \ \underbrace{(\mathsf{D}_1 \,\&\, \ldots \,\&\, \mathsf{D}_k)}_{\text{матрица}} \ , \text{ где}$$

- $Q_1, \ldots, Q_n \in \{\forall, \exists\}$
- матрица — это формула без кванторов в конъюнктивной нормальной форме (КНФ):
  - $\mathsf{D}_i = \mathsf{L}^i_1 \vee \cdots \vee \mathsf{L}^i_{m_i}$ — множитель
  - L^i_j — литера: атом или его отрицание

Наряду с «находится в ПНФ» будем говорить «является ПНФ»

Предварённая нормальная форма

Пример: формула

$$\forall x\ \exists y\ \exists z\ \forall u\ (P(x)\ \&\ \neg R(x,u)\ \&(\neg P(y)\vee R(x,z)))$$

находится в **предварённой нормальной форме**:

- кванторная приставка:

$$\forall x\ \exists y\ \exists z\ \forall u$$

- матрица:

$$P(x)\ \&\ \neg R(x,u)\ \&(\neg P(y)\vee R(x,z))$$

  - множители: $P(x)$, $\neg R(x,u)$, $\neg P(y)\vee R(x,z)$
  - литеры: $P(x)$, $\neg R(x,u)$, $\neg P(y)$, $R(x,z)$

Теорема о предварённой нормальной форме

Для любой замкнутой формулы логики предикатов существует равносильная ей предварённая нормальная форма

Доказательство

Согласно теореме о равносильной замене
и тому, что равносильность является отношением эквивалентности,
достаточно описать способ приведения произвольной формулы к ПНФ
при помощи применения основных равносильностей логики предикатов

Шаги приведения сгруппируем в несколько (5) этапов

Проиллюстрируем устройство этапов на конкретном примере:

$$\neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$

Теорема о предварённой нормальной форме

Для любой замкнутой формулы логики предикатов существует равносильная ей предварённая нормальная форма

Доказательство $\neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$

1. *Переименование переменных*

Переименуем связанные переменные так,
чтобы кванторами связывались попарно различные переменные

Для этого приме́ним основные равносильности

$$\forall \mathrm{x}\,\varphi \sim \forall \mathrm{y}\,(\varphi\{\mathrm{x}/\mathrm{y}\}) \qquad \exists \mathrm{x}\,\varphi \sim \exists \mathrm{y}\,(\varphi\{\mathrm{x}/\mathrm{y}\}),$$

каждый раз выбирая «новую» переменную y

$$\neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$
$$\sim$$
$$\neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{z}\,\mathrm{P}(\mathrm{z}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$
$$\sim$$
$$\neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{z}\,\mathrm{P}(\mathrm{z}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{u}\,\mathrm{R}(\mathrm{x},\mathrm{u}))$$

Теорема о предварённой нормальной форме

Для любой замкнутой формулы логики предикатов существует равносильная ей предварённая нормальная форма

Доказательство $\neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{z}\,\mathrm{P}(\mathrm{z}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{u}\,\mathrm{R}(\mathrm{x},\mathrm{u}))$

2. *Удаление импликаций*

Удалим из формулы все импликации
при помощи основной равносильности

$$\varphi \to \psi \sim \neg\varphi \vee \psi$$

$$\neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{z}\,\mathrm{P}(\mathrm{z}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{u}\,\mathrm{R}(\mathrm{x},\mathrm{u}))$$
$$\sim$$
$$\neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\neg\forall \mathrm{z}\,\mathrm{P}(\mathrm{z}) \vee \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{u}\,\mathrm{R}(\mathrm{x},\mathrm{u}))$$
$$\sim$$
$$\neg\exists \mathrm{x}\,(\neg(\mathrm{P}(\mathrm{x})\,\&(\neg\forall \mathrm{z}\,\mathrm{P}(\mathrm{z}) \vee \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))) \vee \exists \mathrm{u}\,\mathrm{R}(\mathrm{x},\mathrm{u}))$$

Теорема о предварённой нормальной форме

Для любой замкнутой формулы логики предикатов существует равносильная ей предварённая нормальная форма

Доказательство $\neg\exists \mathrm{x}\,(\neg(\mathrm{P}(\mathrm{x})\,\&(\neg\forall \mathrm{z}\,\mathrm{P}(\mathrm{z})\vee\exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})))\vee\exists \mathrm{u}\,\mathrm{R}(\mathrm{x},\mathrm{u}))$

3. *Продвижение отрицаний*

Преобразуем формулу так,
чтобы отрицания располагались только непосредственно над атомами

Для этого приме́ним основные равносильности

$$\neg(\varphi\,\&\,\psi)\sim\neg\varphi\vee\neg\psi \qquad \neg(\varphi\vee\psi)\sim\neg\varphi\,\&\,\neg\psi \qquad \neg\neg\varphi\sim\varphi$$

$$\neg\forall \mathrm{x}\,\varphi\sim\exists \mathrm{x}\,\neg\varphi \qquad \neg\exists \mathrm{x}\,\varphi\sim\forall \mathrm{x}\,\neg\varphi$$

$$\neg\exists \mathrm{x}\,(\neg(\mathrm{P}(\mathrm{x})\,\&(\neg\forall \mathrm{z}\,\mathrm{P}(\mathrm{z})\vee\exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})))\vee\exists \mathrm{u}\,\mathrm{R}(\mathrm{x},\mathrm{u}))$$

$$\sim$$

$$\forall \mathrm{x}\,\neg(\neg(\mathrm{P}(\mathrm{x})\,\&(\neg\forall \mathrm{z}\,\mathrm{P}(\mathrm{z})\vee\exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})))\vee\exists \mathrm{u}\,\mathrm{R}(\mathrm{x},\mathrm{u}))$$

$$\sim$$

$$\forall \mathrm{x}\,(\neg\neg(\mathrm{P}(\mathrm{x})\,\&(\neg\forall \mathrm{z}\,\mathrm{P}(\mathrm{z})\vee\exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})))\&\neg\exists \mathrm{u}\,\mathrm{R}(\mathrm{x},\mathrm{u}))$$

$$\sim$$

$$\forall \mathrm{x}\,(\_\mathrm{P}(\mathrm{x})\,\&(\exists \mathrm{z}\,\neg\mathrm{P}(\mathrm{z})\vee\exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))\,\&\,\forall \mathrm{u}\,\neg\mathrm{R}(\mathrm{x},\mathrm{u}))$$

Теорема о предварённой нормальной форме

Для любой замкнутой формулы логики предикатов существует равносильная ей предварённая нормальная форма

Доказательство $\forall x\,(P(x)\,\&(\exists z\,\neg P(z)\vee\exists y\,R(x,y))\,\&\,\forall u\,\neg R(x,u))$

4. *Вынесение кванторов*

Вынесем все кванторы «наружу», собрав их в кванторную приставку

Для этого приме́ним основные равносильности

$$\forall x\,\varphi\,\&\,\psi\sim\forall x\,(\varphi\,\&\,\psi) \qquad \exists x\,\varphi\,\&\,\psi\sim\exists x\,(\varphi\,\&\,\psi) \qquad \chi_1\,\&\,\chi_2\sim\chi_2\,\&\,\chi_1$$
$$\forall x\,\varphi\vee\psi\sim\forall x\,(\varphi\vee\psi) \qquad \exists x\,\varphi\vee\psi\sim\exists x\,(\varphi\vee\psi) \qquad \chi_1\vee\chi_2\sim\chi_2\vee\chi_1$$

После этапов 2, 3 «над» кванторами могут располагаться только $\&$ и $\vee$

После этапа 1 при вынесении квантора за скобки в ψ не встречается x

$$\forall x\,(P(x)\,\&(\exists z\,\neg P(z)\vee\exists y\,R(x,y))\,\&\,\forall u\,\neg R(x,u))$$
$$\sim$$
$$\forall x\,(P(x)\,\&\,\_\exists z\,(\neg P(z)\vee\exists y\,R(x,y))\,\&\,\forall u\,\neg R(x,u))$$
$$\sim$$
$$\forall x\,(\exists z\,(P(x)\,\&\,\_(\neg P(z)\vee\exists y\,R(x,y)))\,\&\,\forall u\,\neg R(x,u))$$
$$\sim\ldots\sim$$
$$\forall x\,\exists z\,\exists y\,\forall u\,(P(x)\,\&(\neg P(z)\vee R(x,y))\,\&\,\neg R(x,u))$$

Теорема о предварённой нормальной форме

Для любой замкнутой формулы логики предикатов существует равносильная ей предварённая нормальная форма

Доказательство $\forall \mathrm{x}\, \exists \mathrm{z}\, \exists \mathrm{y}\, \forall \mathrm{u}\, (\mathrm{P}(\mathrm{x}) \,\&\, (\neg \mathrm{P}(\mathrm{z}) \vee \mathrm{R}(\mathrm{x}, \mathrm{y})) \,\&\, \neg \mathrm{R}(\mathrm{x}, \mathrm{u}))$

5. *Получение КНФ*

С использованием законов булевой алгебры

$$\psi \vee (\chi_1 \,\&\, \chi_2) \sim (\psi \vee \chi_1) \,\&\, (\psi \vee \chi_2) \qquad\qquad \psi \vee \chi \sim \chi \vee \psi$$

булеву часть формулы можно легко преобразовать в КНФ

В рассматриваемом примере никакие преобразования не нужны, а методы приведения произвольной булевой формулы к КНФ (*вроде бы*) должны быть вам уже́ известны

Итог:

после этапа 4 в формуле появляется **кванторная приставка**,
а после этапа 5 «под» приставкой располагается **КНФ**,
то есть получается ПНФ ▼

Теорема о предварённой нормальной форме

Напоследок — сквозной пример из доказательства от начала до конца

$$\neg\exists x\,(P(x)\,\&(\forall x\,P(x)\to\exists y\,R(x,y))\to\exists y\,R(x,y))$$

$$\neg\exists x\,(P(x)\,\&(\forall z\,P(z)\to\exists y\,R(x,y))\to\exists y\,R(x,y))$$

$$\neg\exists x\,(P(x)\,\&(\forall z\,P(z)\to\exists y\,R(x,y))\to\exists u\,R(x,u))$$

$$\neg\exists x\,(P(x)\,\&(\neg\forall z\,P(z)\vee\exists y\,R(x,y))\to\exists u\,R(x,u))$$

$$\neg\exists x\,(\neg(P(x)\,\&(\neg\forall z\,P(z)\vee\exists y\,R(x,y)))\vee\exists u\,R(x,u))$$

$$\forall x\,\neg(\neg(P(x)\,\&(\neg\forall z\,P(z)\vee\exists y\,R(x,y)))\vee\exists u\,R(x,u))$$

$$\forall x\,(\neg\neg(P(x)\,\&(\neg\forall z\,P(z)\vee\exists y\,R(x,y)))\&\neg\exists u\,R(x,u))$$

$$\forall x\,(\_\,P(x)\,\&(\exists z\,\neg P(z)\vee\exists y\,R(x,y))\,\&\,\forall u\,\neg R(x,u))$$

$$\forall x\,(P(x)\,\&\,\_\,\exists z\,(\neg P(z)\vee\exists y\,R(x,y))\,\&\,\forall u\,\neg R(x,u))$$

$$\forall x\,(\exists z\,(P(x)\,\&\,\_\,(\neg P(z)\vee\exists y\,R(x,y)))\,\&\,\forall u\,\neg R(x,u))$$

...

$$\forall x\,\exists z\,\exists y\,\forall u\,(P(x)\,\&(\neg P(z)\vee R(x,y))\,\&\,\neg R(x,u))$$

Математическая логика и логическое программирование

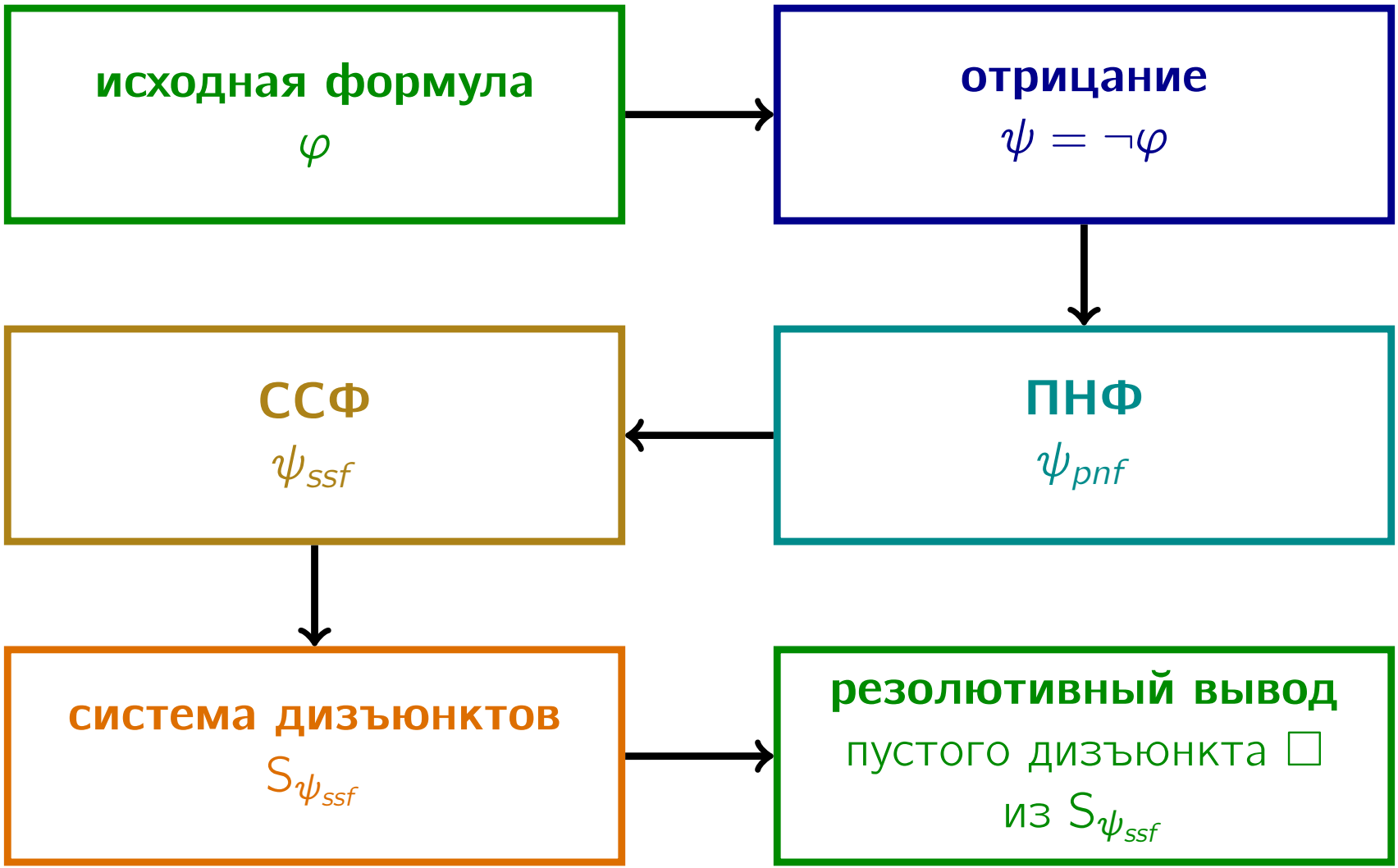
mk.cs.msu.ru → Лекционные курсы
→ Математическая логика и логическое программирование (3-й поток)

Блок 17

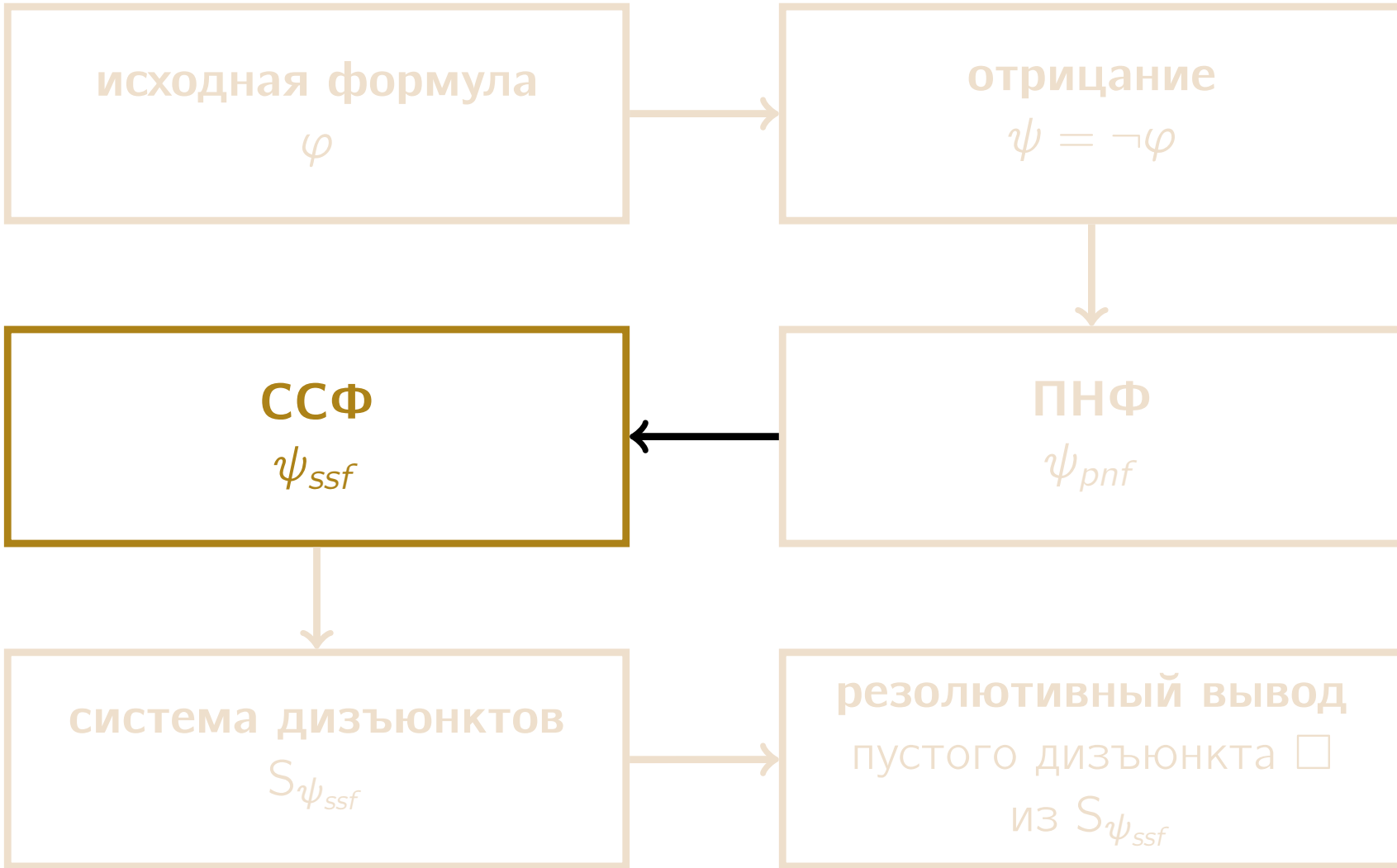
Сколемовская стандартная форма (ССФ)

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

ВМК МГУ, 2023/2024, осенний семестр



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf}$$



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf}$$

Сколемовская стандартная форма

Замкнутая формула логики предикатов находится в сколемовской стандартной форме (ССФ), если

- она находится в предварённой нормальной форме и
- её кванторная приставка не содержит кванторов $\exists$:

$$\forall \widetilde{\mathrm{x}^{\mathrm{n}}}\,(\mathrm{D}_1 \,\&\, \ldots \,\&\, \mathrm{D}_k)$$

Например, формула

$$\forall \mathrm{x}\,\forall \mathrm{u}\,(\mathrm{P}(\mathrm{x}) \,\&\, (\neg \mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x}, \mathbf{g}(\mathrm{x}))) \,\&\, \neg \mathrm{R}(\mathrm{x}, \mathrm{u}))$$

находится в сколемовской стандартной форме

Наряду с «находится в ССФ» будем говорить «является ССФ»

На одном из этапов метода резолюций следует преобразовать ПНФ в настолько же (не)выполнимую ССФ

Лемма об удалении квантора существования

Пусть $\varphi = \forall \widetilde{\mathtt{x}}^{\mathtt{n}}\ \exists \mathtt{x}_{\mathtt{n}+1}\ \chi$ — замкнутая формула ЛП ($n \geq 0$) и функциональный символ $\mathbf{f}$ не содержится в χ. Тогда

$$|\!\models \varphi \quad \Leftrightarrow \quad |\!\models \forall \widetilde{\mathtt{x}}^{\mathtt{n}}\ (\chi\{\mathtt{x}_{\mathtt{n}+1}/\mathbf{f}(\widetilde{\mathtt{x}}^{\mathtt{n}})\})$$

Небольшая вольность: если слева от $\exists$ не стоит ни одного $\forall$, то, согласно лемме, $\mathbf{f}$ — 0-местный функциональный символ: так будем называть **константы**, и писать «$\mathbf{f}()$» наряду с «$\mathbf{f}$»

Доказательство ($\Leftarrow$)

Рассмотрим произвольную модель $\mathcal{I}$ для формулы $\forall \widetilde{\mathtt{x}}^{\mathtt{n}}\ (\chi\{\mathtt{x}_{\mathtt{n}+1}/\mathbf{f}(\widetilde{\mathtt{x}}^{\mathtt{n}})\})$

По семантике «$\forall$», для любого набора предметов $\widetilde{d}^n$ верно

$$\mathcal{I} \models \chi\{\mathtt{x}_{\mathtt{n}+1}/\mathbf{f}(\widetilde{\mathtt{x}}^{\mathtt{n}})\}[\widetilde{d}^n]$$

Следовательно, верно и $\mathcal{I} \models \chi[\widetilde{d}^n, \bar{\mathbf{f}}(\widetilde{d}^n)]$

По семантике «$\exists$», верно $\mathcal{I} \models (\exists \mathtt{x}_{\mathtt{n}+1}\ \chi)[\widetilde{d}^n]$

По семантике «$\forall$», так как набор предметов $\widetilde{d}^n$ произволен, верно и $\mathcal{I} \models \forall \widetilde{\mathtt{x}}^{\mathtt{n}}\ \exists \mathtt{x}_{\mathtt{n}+1}\ \chi$

Лемма об удалении квантора существования

Пусть $\varphi = \forall \widetilde{\mathtt{x}}^{\mathtt{n}}\ \exists \mathtt{x}_{\mathtt{n}+1}\ \chi$ — замкнутая формула ЛП ($n \geq 0$) и функциональный символ $\mathbf{f}$ не содержится в χ. Тогда

$$|\!\models \varphi \quad \Leftrightarrow \quad |\!\models \forall \widetilde{\mathtt{x}}^{\mathtt{n}}\ (\chi\{\mathtt{x}_{\mathtt{n}+1}/\mathbf{f}(\widetilde{\mathtt{x}}^{\mathtt{n}})\})$$

Доказательство ($\Rightarrow$)

Рассмотрим произвольную модель $\mathcal{I}$ для формулы φ

По семантике «$\forall$» и «$\exists$», для любого набора предметов $\widetilde{d}^n$ существует предмет d_{n+1}, такой что $\mathcal{I} \models \chi[\widetilde{d}^n, d_{n+1}]$

По $\mathcal{I}$ построим **новую** интерпретацию $\mathcal{J}$:

- если в сигнатуре был функциональный символ $\mathbf{f}$, удалим его
- добавим в сигнатуру функциональный символ $\mathbf{f}^{(n)}$
- оценим $\mathbf{f}$ так, чтобы выполнялось равенство $\bar{\mathbf{f}}(\widetilde{d}^n) = d_{n+1}$

Тогда $\mathcal{J} \models \chi[\widetilde{d}^n, \bar{\mathbf{f}}(\widetilde{d}^n)]$, а значит, и $\mathcal{J} \models \chi\{\mathtt{x}_{\mathtt{n}+1}/\mathbf{f}(\widetilde{\mathtt{x}}^{\mathtt{n}})\}[\widetilde{d}^n]$

По семантике «$\forall$», так как набор предметов $\widetilde{d}^n$ произволен, верно и $\mathcal{J} \models \forall \widetilde{\mathtt{x}}^{\mathtt{n}}\ (\chi\{\mathtt{x}_{\mathtt{n}+1}/\mathbf{f}(\widetilde{\mathtt{x}}^{\mathtt{n}})\})$ ▼

Лемма об удалении квантора существования

Пусть $\varphi = \forall \widetilde{\mathtt{x}}^{\mathtt{n}}\ \exists \mathtt{x}_{\mathtt{n}+1}\ \chi$ — замкнутая формула ЛП ($n \geq 0$) и функциональный символ f не содержится в χ. Тогда

$$|\!\models \varphi \quad \Leftrightarrow \quad |\!\models \forall \widetilde{\mathtt{x}}^{\mathtt{n}}\ (\chi\{\mathtt{x}_{\mathtt{n}+1}/\mathbf{f}(\widetilde{\mathtt{x}}^{\mathtt{n}})\})$$

Устранение квантора $\exists$ с введением новых символов с целью получить более простую «хорошую» формулу обычно называют сколемизацией (*здесь «хорошая» — сохраняющая выполнимость и невыполнимость*)

При устранении $\exists$ на место удаляемой переменной подставляются сколемовские термы (*здесь* — $\mathbf{f}(\widetilde{\mathtt{x}}^{\mathtt{n}})$)

Алгоритм сколемизации ПНФ

Дано: ПНФ φ_{pnf}

Требуется получить ССФ $Sk(\varphi_{pnf})$, такую что

$$\varphi_{pnf} \text{ выполнима} \quad \Leftrightarrow \quad Sk(\varphi_{pnf}) \text{ выполнима}$$

Алгоритм. Пока в кванторной приставке есть хотя бы один квантор $\exists$, самый левый $\exists$ удаляется при помощи подстановки сколемовского терма:

$$\varphi_{pnf} = \forall \mathtt{x}_1 \ \ldots \forall \mathtt{x}_{k-1}\ \exists \mathtt{x}_k\ \forall \mathtt{x}_{k+1}\ \ldots \forall \mathtt{x}_{m-1}\ \exists \mathtt{x}_m\ \ldots\ \chi$$

$$\forall \mathtt{x}_1 \ \ldots \forall \mathtt{x}_{k-1}\ \forall \mathtt{x}_{k+1}\ \ldots \forall \mathtt{x}_{m-1}\ \exists \mathtt{x}_m\ \ldots\ (\chi\{\mathtt{x}_k/\mathbf{f}_1(\mathtt{x}_1, \ldots, \mathtt{x}_{k-1})\})$$

$$\forall \mathtt{x}_1 \ \ldots \forall \mathtt{x}_{k-1}\ \forall \mathtt{x}_{k+1}\ \ldots \forall \mathtt{x}_{m-1}\ \ldots \\ (\chi\{\mathtt{x}_k/\mathbf{f}_1(\mathtt{x}_1, \ldots, \mathtt{x}_{k-1}), \mathtt{x}_m/\mathbf{f}_2(\mathtt{x}_1, \ldots, \mathtt{x}_{k-1}, \mathtt{x}_{k+1}, \ldots, \mathtt{x}_{m-1})\})$$

$$\ldots$$

$$Sk(\varphi_{pnf})$$

При этом **важно** (**!**) при удалении очередного $\exists$
каждый раз выбирать **новый** функциональный символ:
не содержащийся в формуле после всех предыдущих удалений

Алгоритм сколемизации ПНФ

Пример:

$$\forall \mathrm{x}\ \exists \mathrm{z}\ \exists \mathrm{y}\ \forall \mathrm{u}\ (\mathrm{P}(\mathrm{x}) \,\&\, (\neg \mathrm{P}(\mathrm{z}) \vee \mathrm{R}(\mathrm{x}, \mathrm{y})) \,\&\, \neg \mathrm{R}(\mathrm{x}, \mathrm{u}))$$

$$\forall \mathrm{x}\ \exists \mathrm{y}\ \forall \mathrm{u}\ (\mathrm{P}(\mathrm{x}) \,\&\, (\neg \mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x}, \mathrm{y})) \,\&\, \neg \mathrm{R}(\mathrm{x}, \mathrm{u}))$$

$$\forall \mathrm{x}\ \forall \mathrm{u}\ (\mathrm{P}(\mathrm{x}) \,\&\, (\neg \mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x}, \mathbf{g}(\mathrm{x}))) \,\&\, \neg \mathrm{R}(\mathrm{x}, \mathrm{u}))$$

Ограничения на выбор символов в сколемовских термах:

- **f** — любой функциональный символ местности 1 (ни один не содержится в формуле)
- **g** — любой функциональный символ местности 1, кроме **f**

Алгоритм сколемизации ПНФ

Другой пример:

$$\exists u\ \exists v\ \forall w\ \exists x\ \forall y\ \exists z\ (P(u, v) \,\&\, P(v, w) \,\&\, P(w, x) \,\&\, P(x, y) \,\&\, P(y, z) \,\&\, P(z, u))$$

$$\exists v\ \forall w\ \exists x\ \forall y\ \exists z\ (P(\mathbf{c}, v) \,\&\, P(v, w) \,\&\, P(w, x) \,\&\, P(x, y) \,\&\, P(y, z) \,\&\, P(z, \mathbf{c}))$$

$$\forall w\ \exists x\ \forall y\ \exists z\ (P(\mathbf{c}, \mathbf{d}) \,\&\, P(\mathbf{d}, w) \,\&\, P(w, x) \,\&\, P(x, y) \,\&\, P(y, z) \,\&\, P(z, \mathbf{c}))$$

$$\forall w\ \forall y\ \exists z\ (P(\mathbf{c}, \mathbf{d}) \,\&\, P(\mathbf{d}, w) \,\&\, P(w, \mathbf{f}(w)) \,\&\, P(\mathbf{f}(w), y) \,\&\, P(y, z) \,\&\, P(z, \mathbf{c}))$$

$$\forall w\ \forall y\ (\ P(\mathbf{c}, \mathbf{d}) \,\&\, P(\mathbf{d}, w) \,\&\, P(w, \mathbf{f}(w)) \,\&\, P(\mathbf{f}(w), y) \,\&\, P(y, \mathbf{g}(w, y)) \,\&\, P(\mathbf{g}(w, y), \mathbf{c})\ )$$

Ограничения на выбор символов в сколемовских термах:

- **c** — любая константа (ни одной не содержится в формуле)
- **d** — любая константа, кроме **c**
- **f** — любой функциональный символ местности 1 (ни один не содержится в формуле)
- **g** — любой функциональный символ местности 2 (ни один не содержится в формуле)

Алгоритм сколемизации ПНФ

Теорема (о сколемизации). Для любой ПНФ φ_{pnf} формула $Sk(\varphi_{pnf})$ является ССФ, для которой верно следующее:

$$|\models \varphi_{pnf} \quad \Leftrightarrow \quad |\models Sk(\varphi_{pnf})$$

Доказательство.

Пусть $\psi_1, \psi_2, \ldots, \psi_k$ — формулы, последовательно получающиеся удалением кванторов $\exists$ (по одному) согласно алгоритму сколемизации ($\psi_1 = \varphi_{pnf}$, $\psi_k = Sk(\varphi_{pnf})$)

По лемме об удалении квантора $\exists$, справедливы равносильности

$$|\models \varphi_{pnf} \quad \Leftrightarrow \quad |\models \psi_2 \quad \Leftrightarrow \ldots \Leftrightarrow \quad |\models Sk(\varphi_{pnf}) \ \blacktriangledown$$

А если в формулировке теоремы заменить «выполнима» ($|\models$) на «общезначима» ($\models$), останутся ли верными $\Rightarrow$ и $\Leftarrow$?

Математическая логика и логическое программирование

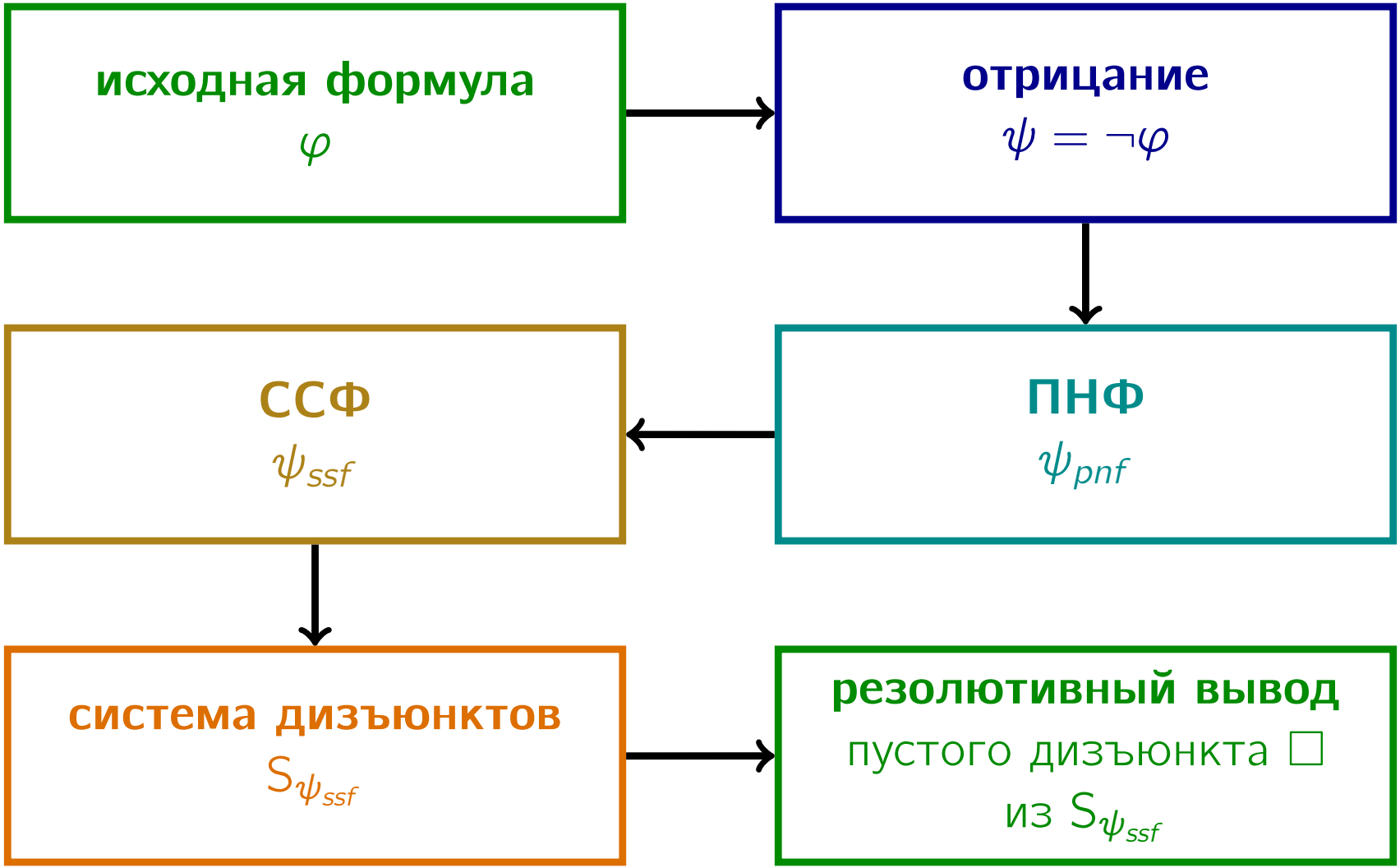
mk.cs.msu.ru → Лекционные курсы
→ Математическая логика и логическое программирование (3-й поток)

Блок 18

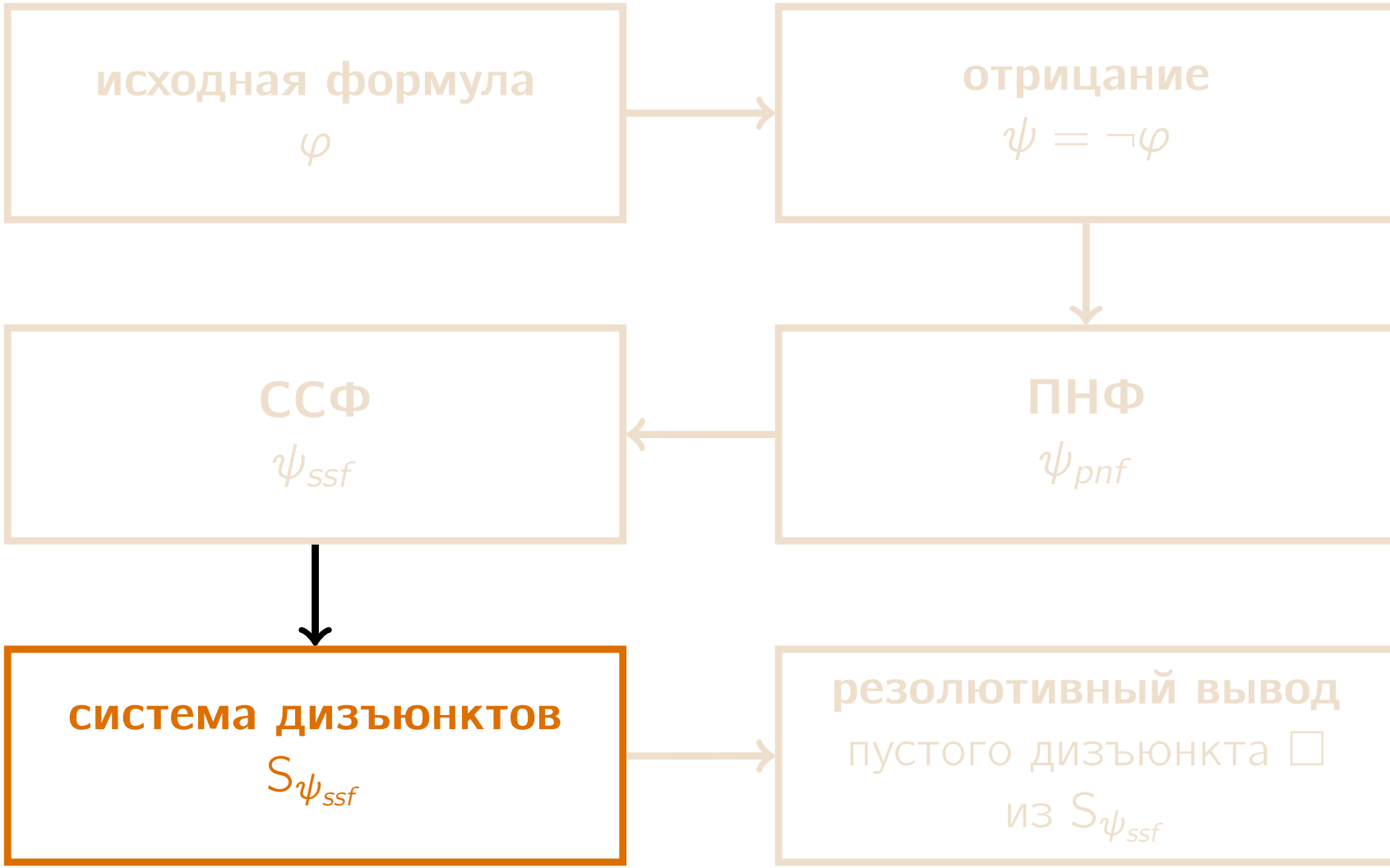
Системы дизъюнктов

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

ВМК МГУ, 2023/2024, осенний семестр



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf}$$



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf}$$

Системы дизъюнктов

Дизъюнктом называется ССФ с одним множителем в матрице:

$$\forall\widetilde{\mathrm{x}}^{\mathrm{n}}\,(\mathrm{L}_1 \vee \cdots \vee \mathrm{L}_k),$$

где L_i — литера (атом или его отрицание)

Для краткости иногда будем опускать кванторную приставку дизъюнктов:

$$\forall\widetilde{\mathrm{x}}^{\mathrm{n}}\,(\mathrm{L}_1 \vee \cdots \vee \mathrm{L}_k) = \mathrm{L}_1 \vee \cdots \vee \mathrm{L}_k$$

Для упрощения технических выкладок будем отождествлять между собой дизъюнкты, получающиеся друг из друга перестановкой слагаемых

В связи с таким упрощением будем отождествлять дизъюнкт с мультимножеством его литер:

$$\mathrm{L}_1 \vee \cdots \vee \mathrm{L}_k = \{\mathrm{L}_1, \ldots, \mathrm{L}_k\}$$

Системы дизъюнктов

Например: (*но это* **только** *при обсуждении дизъюнктов*)

$$\forall \mathrm{x}\,(\mathrm{P}(\mathrm{x}) \vee \mathrm{P}(\mathrm{x}) \vee \mathrm{Q}(\mathbf{f}(\mathbf{c})))$$

$$=$$

$$\mathrm{P}(\mathrm{x}) \vee \mathrm{P}(\mathrm{x}) \vee \mathrm{Q}(\mathbf{f}(\mathbf{c}))$$

$$=$$

$$\{\mathrm{P}(\mathrm{x}), \mathrm{P}(\mathrm{x}), \mathrm{Q}(\mathbf{f}(\mathbf{c}))\}$$

$$=$$

$$\mathrm{P}(\mathrm{x}) \vee \mathrm{Q}(\mathbf{f}(\mathbf{c})) \vee \mathrm{P}(\mathrm{x})$$

$$=$$

$$\forall \mathrm{x}\,(\mathrm{P}(\mathrm{x}) \vee \mathrm{Q}(\mathbf{f}(\mathbf{c})) \vee \mathrm{P}(\mathrm{x}))$$

Пустой дизъюнкт $\square$ — это особый дизъюнкт, представляющий собой пустое множество литер

Пустой дизъюнкт будем считать **невыполнимым**:

$$\mathsf{L}_1 \vee \cdots \vee \mathsf{L}_k \text{ «}\sim\text{» } \mathsf{L}_1 \vee \cdots \vee \mathsf{L}_k \vee \mathbb{f}, \text{ а значит, } \square \text{ «}\sim\text{» } \mathbb{f}$$

Системой дизъюнктов будем называть (любое) множество дизъюнктов

Системы дизъюнктов

Утверждение. $\forall \mathrm{x}\,(\varphi \,\&\, \psi) \sim \forall \mathrm{x}\,\varphi \,\&\, \forall \mathrm{x}\,\psi$

Доказательство. Очевидно?

(*Обосновать эту равносильность настолько же просто, как и все основные равносильности*)

Теорема (о переходе к дизъюнктам)

Для ССФ с любым набором множителей $\mathrm{D}_1, \ldots, \mathrm{D}_k$ **верно:**

$$|\!\models \forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,(\mathrm{D}_1 \,\&\, \ldots \,\&\, \mathrm{D}_k) \quad \Leftrightarrow \quad |\!\models \{\forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,\mathrm{D}_1, \ldots, \forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,\mathrm{D}_k\}$$

Доказательство

По утверждению выше, $\forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,(\mathrm{D}_1 \,\&\, \ldots \,\&\, \mathrm{D}_k) \sim \forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,\mathrm{D}_1 \,\&\, \ldots \,\&\, \forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,\mathrm{D}_k$

Следовательно, с учётом семантики &,

$|\!\models \forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,(\mathrm{D}_1 \,\&\, \ldots \,\&\, \mathrm{D}_k)$

$\Leftrightarrow |\!\models \forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,\mathrm{D}_1 \,\&\, \ldots \,\&\, \forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,\mathrm{D}_k$

$\Leftrightarrow |\!\models \{\forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,\mathrm{D}_1, \ldots, \forall\widetilde{\mathrm{x}^{\mathrm{n}}}\,\mathrm{D}_k\}$ ▼

Системы дизъюнктов

Пример:

$$\not\models \forall \mathrm{x}\, \forall \mathrm{u}\, (\mathrm{P}(\mathrm{x}) \,\&\, (\neg \mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x}, \mathbf{g}(\mathrm{x}))) \,\&\, \neg \mathrm{R}(\mathrm{x}, \mathrm{u}))$$

$$\Leftrightarrow$$

$$\not\models \{\mathrm{P}(\mathrm{x}), \qquad \neg \mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x}, \mathbf{g}(\mathrm{x})), \qquad \neg \mathrm{R}(\mathrm{x}, \mathrm{u})\}$$

Математическая логика и логическое программирование

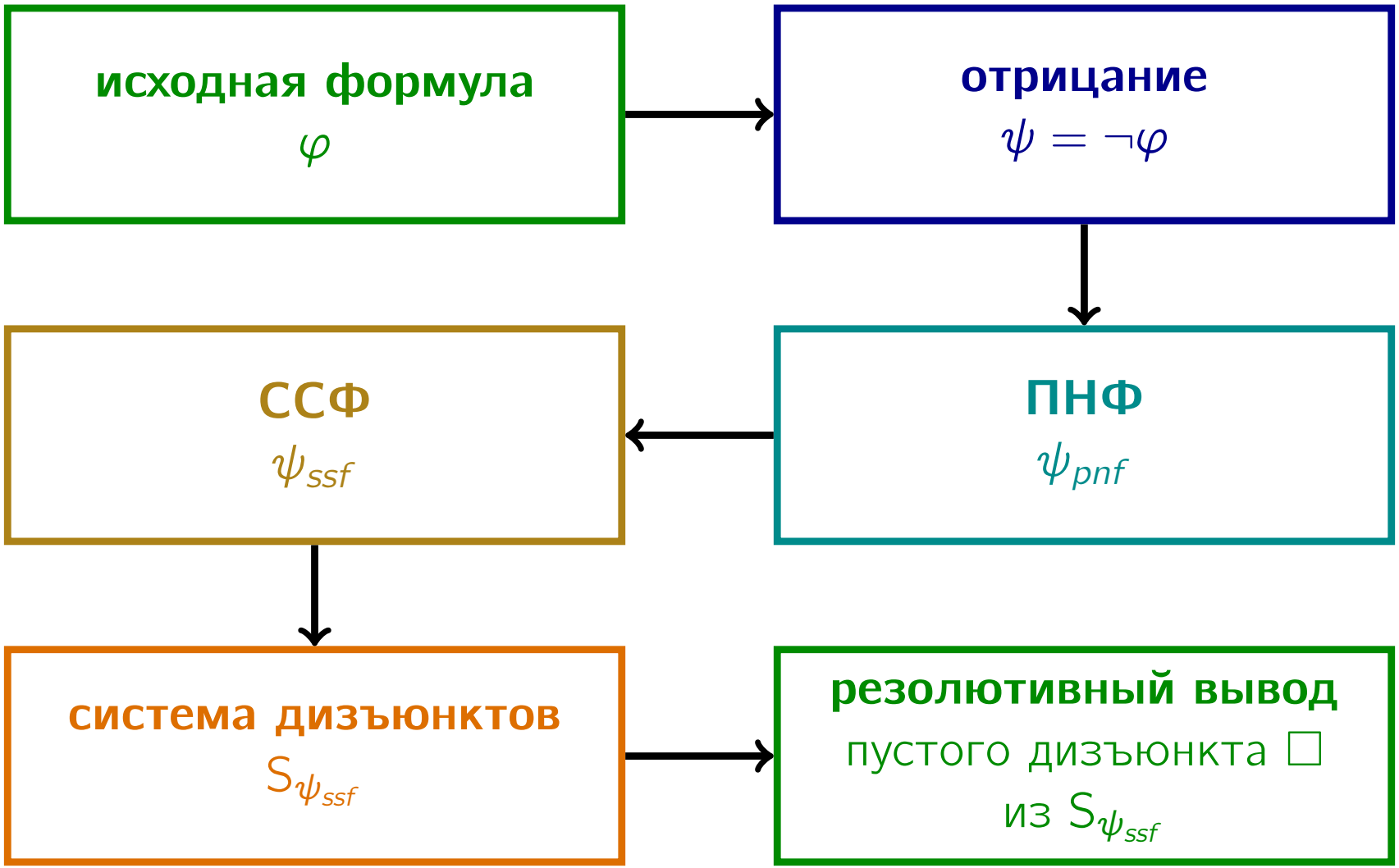
mk.cs.msu.ru → Лекционные курсы
→ Математическая логика и логическое программирование (3-й поток)

Блок 19

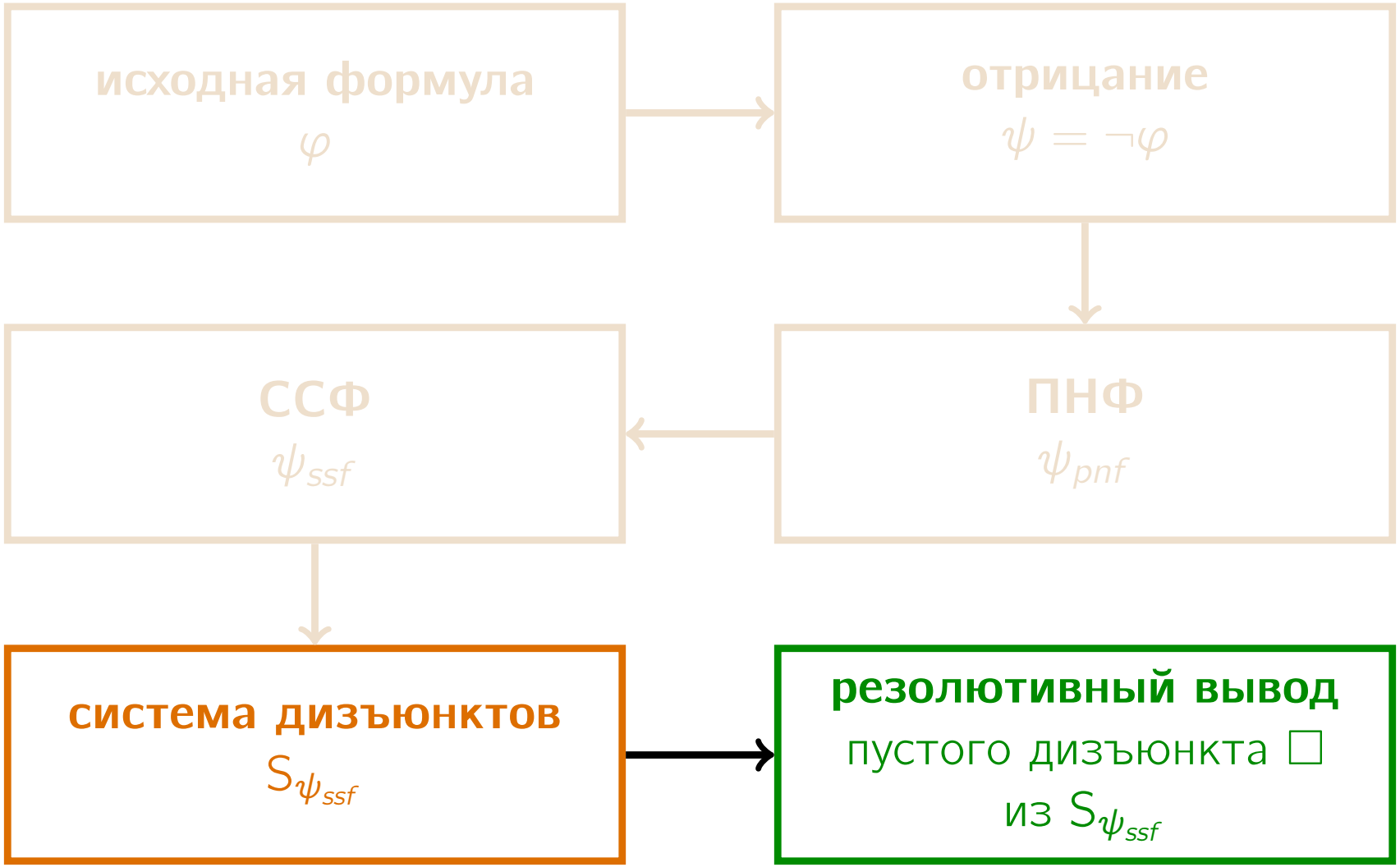
Композиция подстановок
Постановка задачи унификации

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

ВМК МГУ, 2023/2024, осенний семестр



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf} \quad \Leftrightarrow \quad \not\models S_{\psi_{ssf}}$$



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf} \quad \Leftrightarrow \quad \not\models S_{\psi_{ssf}}$$

Противоречия в системах дизъюнктов

На последних этапах метода резолюций
требуется проверить выполнимость системы дизъюнктов

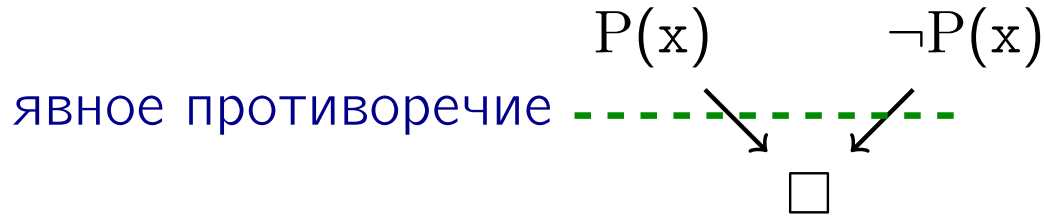
Попробуем, по аналогии с методом семантических таблиц,
придумать способ извлечения явных противоречий
из скрытых (неявных)

В качестве явного противоречия
будем использовать пустой дизъюнкт ($\square$)

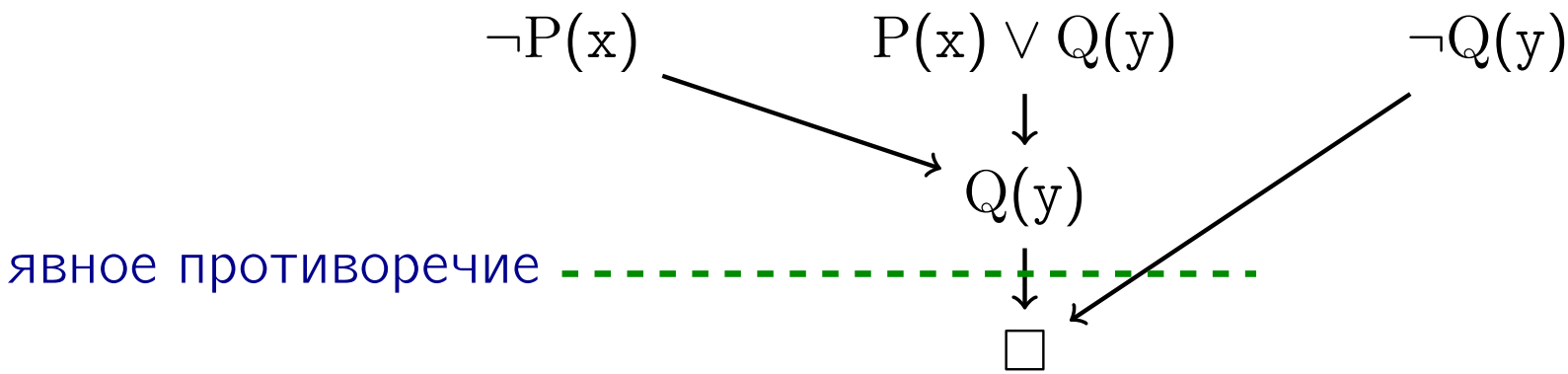
Начнём издалека: приведём несколько примеров того,
как можно было бы «разумно» и «просто» извлечь $\square$
из невыполнимой системы дизъюнктов

Противоречия в системах дизъюнктов

$\not\models \{\mathrm{P(x)},\ \neg\mathrm{P(x)}\}$:



$\not\models \{\neg\mathrm{P(x)},\ \neg\mathrm{Q(y)},\ \mathrm{P(x)} \vee \mathrm{Q(y)}\}$:

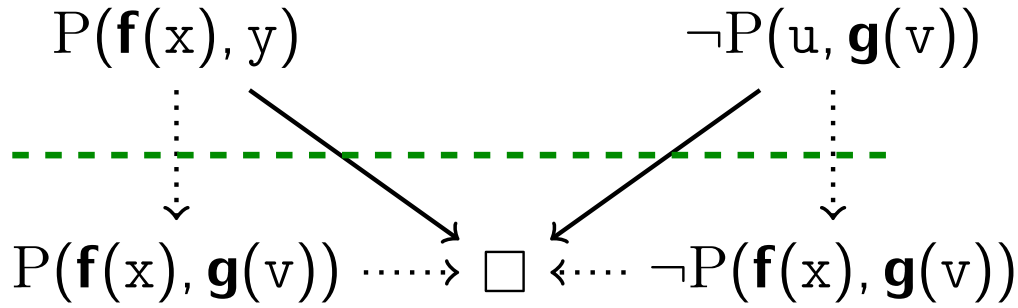


$$\forall \mathrm{x}\ \neg\mathrm{P(x)},\ \forall \mathrm{x}\ \forall \mathrm{y}\ (\mathrm{P(x)} \vee \mathrm{Q(y)}) \models \forall \mathrm{y}\ \mathrm{Q(y)}$$

Противоречия в системах дизъюнктов

$\not\models \{\mathrm{P}(\mathbf{f}(\mathrm{x}), \mathrm{y}),\ \neg\mathrm{P}(\mathrm{u}, \mathbf{g}(\mathrm{v}))\}$

не очень явное противоречие



$$\forall \mathrm{x}\ \forall \mathrm{y}\ \mathrm{P}(\mathbf{f}(\mathrm{x}), \mathrm{y}) \models \forall \mathrm{x}\ \forall \mathrm{v}\ \mathrm{P}(\mathbf{f}(\mathrm{x}), \mathbf{g}(\mathrm{v}))$$

$$\forall \mathrm{u}\ \forall \mathrm{v}\ \neg\mathrm{P}(\mathrm{u}, \mathbf{g}(\mathrm{v})) \models \forall \mathrm{x}\ \forall \mathrm{v}\ \neg\mathrm{P}(\mathbf{f}(\mathrm{x}), \mathbf{g}(\mathrm{v}))$$

Чтобы обнаружить не очень явное противоречие в системе дизъюнктов, потребовалось привести атомы дизъюнктов **к общему виду**

Приведение выражений к общему виду называется унификацией, и перед описанием последнего этапа метода резолюций (**резолютивного вывода** $\square$) необходимо строго сформулировать и научиться решать эту задачу

Композиция подстановок

Унификация атомов A, B достигается применением к ним подстановки θ, такой что $A\theta = B\theta$

Напоминание

Подстановка — это отображение $\theta : \mathrm{Var} \to \mathrm{Term}$

Конечная подстановка задаётся множеством связок:

$$\{\mathtt{x}_1/t_1,\ \ldots,\ \mathtt{x}_\mathtt{n}/t_n\}$$

$E\theta$ — это результат применения подстановки θ к выражению E

Чтобы поставить и решить задачу унификации, исследуем основные алгебраические свойства подстановок

Композиция подстановок

Композиция подстановок θ и η — это подстановка $\theta\eta$, такая что для любой переменной x верно равенство

$$\mathrm{x}(\theta\eta) = (\mathrm{x}\theta)\eta$$

Утверждение

Пусть $\theta = \{\mathrm{x}_1/t_1, \ldots, \mathrm{x}_\mathrm{n}/t_n\}$ **и** $\eta = \{\mathrm{y}_1/s_1, \ldots, \mathrm{y}_\mathrm{k}/s_k\}$**. Тогда**

$$\theta\eta = \{\mathrm{x}_\mathrm{i}/t_i\eta \mid 1 \le i \le n,\ x_i \neq t_i\eta\} \cup \{\mathrm{y}_\mathrm{j}/s_j \mid 1 \le j \le k,\ \mathrm{y}_\mathrm{j} \notin \{\mathrm{x}_1, \ldots, \mathrm{x}_\mathrm{n}\}\}$$

Доказательство

Рассмотрим переменную $\mathrm{z} \in \mathrm{Var}$

Если $\mathrm{z} \notin \mathrm{Dom}_\theta \cup \mathrm{Dom}_\eta$, то $\mathrm{z}(\theta\eta) = (\mathrm{z}\theta)\eta = \mathrm{z}\eta = \mathrm{z}$

Если $\mathrm{z} = \mathrm{y}_\mathrm{j} \in \mathrm{Dom}_\eta \setminus \mathrm{Dom}_\theta$, то $\mathrm{z}(\theta\eta) = (\mathrm{z}\theta)\eta = \mathrm{z}\eta = s_j$

Иначе $\mathrm{z} = \mathrm{x}_\mathrm{i} \in \mathrm{Dom}_\theta$, и $\mathrm{z}(\theta\eta) = (\mathrm{z}\theta)\eta = t_i\eta$ ▼

Композиция подстановок

Пример

$\theta = \{\mathrm{x}/\mathbf{f}(\mathrm{x}, \mathbf{c}),\ \mathrm{y}/\mathbf{g}(\mathrm{u}),\ \mathrm{z}/\mathrm{y}\}$

$\eta = \{\mathrm{x}/\mathbf{g}(\mathrm{y}), \mathrm{y}/\mathrm{z}, \mathrm{u}/\mathbf{c}\}$

$\theta\eta = ?$

$$\{\mathrm{x}/\mathbf{f}(\mathrm{x}, \mathbf{c})\eta,\ \mathrm{y}/\mathbf{g}(\mathrm{u})\eta,\ \mathrm{z}/\mathrm{y}\eta\} \cup \{\mathrm{u}/\mathbf{c}\}$$

$$\{\mathrm{x}/\mathbf{f}(\mathbf{g}(\mathrm{y}), \mathbf{c}),\ \mathrm{y}/\mathbf{g}(\mathbf{c}),\ \mathrm{z}/\mathrm{z}\} \cup \{\mathrm{u}/\mathbf{c}\}$$

$$\{\mathrm{x}/\mathbf{f}(\mathbf{g}(\mathrm{y}), \mathbf{c}),\ \mathrm{y}/\mathbf{g}(\mathbf{c})\} \cup \{\mathrm{u}/\mathbf{c}\}$$

$$\theta\eta = \{\mathrm{x}/\mathbf{f}(\mathbf{g}(\mathrm{y}), \mathbf{c}),\ \mathrm{y}/\mathbf{g}(\mathbf{c}),\ \mathrm{u}/\mathbf{c}\}$$

Задача унификации

Подстановка θ называется унификатором выражений E_1, E_2, если $E_1\theta = E_2\theta$

$\text{У}(E_1, E_2)$ — множество всех унификаторов выражений E_1, E_2

Выражения E_1, E_2 унифицируемы, если $\text{У}(E_1, E_2) \neq \emptyset$

Утверждение

Для любых подстановок θ, η и любых выражений E_1, E_2 верно:

$$\textbf{если } \theta \in \textbf{У}(E_1, E_2)\textbf{, то } \theta\eta \in \textbf{У}(E_1, E_2)$$

Доказательство

$$\theta \in \text{У}(E_1, E_2) \quad \Leftrightarrow \quad E_1\theta = E_2\theta \quad \Rightarrow \quad E_1\theta\eta = E_2\theta\eta \quad \Leftrightarrow \quad \theta\eta \in \text{У}(E_1, E_2)$$ ▼

Подмножество S множества подстановок Θ называется полным в Θ, если любая подстановка θ из Θ представима в виде $\theta = \eta\mu$, где $\eta \in S$

Подстановка θ называется наиболее общим унификатором выражений E_1, E_2, если множество $\{\theta\}$ является полным в $\text{У}(E_1, E_2)$

$\text{НОУ}(E_1, E_2)$ — множество всех наиболее общих унификаторов выражений E_1, E_2

Задача унификации

Пример

Рассмотрим два атома:

$$A = \mathrm{P}(\mathbf{f}(\mathrm{x}), \mathrm{y}) \qquad B = \mathrm{P}(\mathrm{u}, \mathbf{g}(\mathrm{v}))$$

Подстановка $\eta = \{\mathrm{y}/\mathbf{g}(\mathbf{g}(\mathrm{v})), \mathrm{u}/\mathbf{f}(\mathbf{c}), \mathrm{v}/\mathbf{g}(\mathrm{v}), \mathrm{x}/\mathbf{c}\}$ — унификатор атомов A и B (то есть $\eta \in \text{У}(A, B)$), т.к.

$$\mathrm{P}(\mathbf{f}(\mathrm{x}), \mathrm{y})\eta = \mathrm{P}(\mathbf{f}(\mathbf{c}), \mathbf{g}(\mathbf{g}(\mathrm{v}))) = \mathrm{P}(\mathrm{u}, \mathbf{g}(\mathrm{v}))\eta$$

А подстановка $\theta = \{\mathrm{y}/\mathbf{g}(\mathrm{v}), \mathrm{u}/\mathbf{f}(\mathrm{x})\}$ — *более общий* унификатор A и B:

$$\mathrm{P}(\mathbf{f}(\mathrm{x}), \mathrm{y})\theta = \mathrm{P}(\mathbf{f}(\mathrm{x}), \mathbf{g}(\mathrm{v})) = \mathrm{P}(\mathrm{u}, \mathbf{g}(\mathrm{v}))\theta$$

$$\eta = \theta\{\mathrm{v}/\mathbf{g}(\mathrm{v}), \mathrm{x}/\mathbf{c}\}$$

На самом деле θ — **наиболее общий унификатор** атомов A и B

(но как это доказать?)

А выражения $\mathrm{P}(\mathrm{x}, \mathbf{f}(\mathrm{x}))$ и $\mathrm{P}(\mathbf{g}(\mathrm{y}), \mathrm{y})$ **неунифицируемы**

(а это как доказать?)

Задача унификации

формулируется следующим образом:

для заданных выражений E_1, E_2

выяснить, унифицируемы ли эти выражения,

и если это так, то вычислить

множество унификаторов, полное в $\mathbf{У}(E_1, E_2)$

Математическая логика и логическое программирование

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

Блок 20

Алгоритм унификации атомарных формул

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

ВМК МГУ, 2023/2024, осенний семестр

Напоминание: задача унификации

Для заданных выражений E_1, E_2 логики предикатов

выяснить, унифицируемы ли эти выражения,

и если это так, то вычислить

множество унификаторов, полное в У(E_1, E_2)

Чтобы освоить метод резолюций, достаточно
научиться решать эту задачу для произвольной пары **атомов**

Переход от атомов к системам уравнений

Утверждение

Никакая пара атомов $\mathrm{P}(t_1, \ldots, t_k)$, $\mathrm{Q}(s_1, \ldots, s_m)$ с различными предикатными символами $\mathrm{P}^{(k)}$, $\mathrm{Q}^{(m)}$ не унифицируема

Доказательство

Пусть $\mathrm{P}(\widetilde{t}^k)\theta = \mathrm{P}(\widetilde{t'}^k)$ и $\mathrm{Q}(\widetilde{s}^m)\theta = \mathrm{Q}(\widetilde{s'}^m)$

Т.к. $\mathrm{P} \neq \mathrm{Q}$, верно и $\mathrm{P}(\widetilde{t'}^k) \neq \mathrm{Q}(\widetilde{s'}^m)$ ▼

Далее рассматривается только унификация атомов, содержащих одинаковые предикатные символы

Переход от атомов к системам уравнений

Унификация атомов $\mathrm{P}(t_1, \ldots, t_k)$, $\mathrm{P}(s_1, \ldots, s_k)$

$$\Leftrightarrow$$

Вычисление подстановки θ, такой что $t_1\theta = s_1\theta, \quad \ldots, \quad t_k\theta = s_k\theta$

$$\Leftrightarrow$$

Вычисление подстановки θ, такой что левая и правая части каждого уравнения в системе $\mathcal{E}$ вида

$$\begin{cases} t_1 = s_1 \\ \ldots \\ t_k = s_k \end{cases}$$

становятся посимвольно одинаковыми при применении θ

$$\Leftrightarrow$$

Вычисление **решения системы уравнений** $\mathcal{E}$
в свободной[1] алгебре термов[2]

---

1 Значение терма — это сам терм,
то есть термы равны, если они посимвольно совпадают

2 Операция композиции — это подстановка терма на место переменной

Системы уравнений над термами

Чтобы отличать равенство в уравнениях системы
от посимвольного совпадения выражений и от других видов равенства,
будем знак равенства в системах уравнений записывать так: $\eqq$

Системой уравнений (*в свободной алгебре термов*) будем называть
запись $\mathcal{E}$ вида

$$\begin{cases} t_1 \eqq s_1 \\ \dots \\ t_k \eqq s_k \end{cases},$$

где $t_1, \dots, t_k, s_1, \dots, s_k$ — термы

Подстановка θ — унификатор (*решение*) системы $\mathcal{E}$,
если для каждого i, $i \in \{1, 2, \dots, k\}$, верно $t_i\theta = s_i\theta$

$\text{У}(\mathcal{E})$ — множество всех унификаторов системы уравнений $\mathcal{E}$

Система уравнений $\mathcal{E}$ унифицируема (*имеет решение*), если $\text{У}(\mathcal{E}) \neq \emptyset$

$\text{НОУ}(\mathcal{E})$ — множество всех наиболее общих унификаторов
системы уравнений $\mathcal{E}$

Системы уравнений над термами

Примеры

Пусть $\mathcal{E} = \begin{cases} \mathbf{f}(\mathbf{c}, \mathrm{x}) \doteq \mathbf{f}(\mathrm{y}, \mathbf{g}(\mathrm{y})) \\ \mathbf{g}(\mathrm{y}) \doteq \mathrm{z} \end{cases}$ и $\theta = \{\mathrm{x}/\mathbf{g}(\mathbf{c}), \mathrm{y}/\mathbf{c}, \mathrm{z}/\mathbf{g}(\mathbf{c})\}$

Тогда $\mathcal{E}\theta = \begin{cases} \mathbf{f}(\mathbf{c}, \mathbf{g}(\mathbf{c})) \doteq \mathbf{f}(\mathbf{c}, \mathbf{g}(\mathbf{c})) \\ \mathbf{g}(\mathbf{c}) \doteq \mathbf{g}(\mathbf{c}) \end{cases}$ и θ — унификатор системы

(А как его вычислить и проверить, наиболее общий ли он?)

А система $\begin{cases} \mathbf{f}(\mathbf{c}, \mathrm{y}) \doteq \mathbf{f}(\mathrm{y}, \mathbf{g}(\mathrm{y})) \\ \mathbf{g}(\mathrm{y}) \doteq \mathrm{z} \end{cases}$ неунифицируема

(А как такое доказать в общем случае?)

Системы уравнений над термами

Утверждение. **Множества унификаторов любой пары атомов**

$$\mathrm{P}(t_1, \ldots, t_k), \mathrm{P}(s_1, \ldots, s_k)$$

и системы уравнений

$$\begin{cases} t_1 \stackrel{?}{=} s_1 \\ \ldots \\ t_k \stackrel{?}{=} s_k \end{cases}$$

совпадают

Доказательство. Очевидным образом следует
из определений унификатора атомов и унификатора системы ▼

Унификация переменной и терма

Лемма (о связке). **Для любых переменной** x **и терма** t **верно:**

1. **Если** $x = t$**, то** $\varepsilon \in$ **НОУ**(x, t)
2. **Если** $\mathrm{x} \notin \mathrm{Var}_t$**, то** $\{\mathrm{x}/t\} \in$ **НОУ**(x, t)
3. **Если** $\mathrm{x} \in \mathrm{Var}_t$ **и** $\mathrm{x} \neq t$**, то У**$(\mathrm{x}, t) = \emptyset$

Доказательство

1. Следует из того, что $\mathrm{x}\varepsilon = \mathrm{x}$ и для любой подстановки η верно $\eta = \varepsilon\eta$

2. Достаточно показать, что если $\mathrm{x} \notin \mathrm{Var}_t$, то:

а) $\{\mathrm{x}/t\}$ — унификатор (*переменной* x *и терма* t)

б) для любого унификатора θ существует подстановка η, такая что

$$\theta = \{\mathrm{x}/t\}\eta$$

а) Следует из равенств $\mathrm{x}\{\mathrm{x}/t\} = t = t\{\mathrm{x}/t\}$

Унификация переменной и терма

Лемма (о связке). **Для любых переменной** x **и терма** t **верно:**

1. **Если** $x = t$**, то** $\varepsilon \in$ **НОУ**(x, t)
2. **Если** $\mathrm{x} \notin \mathrm{Var}_t$**, то** $\{\mathrm{x}/t\} \in$ **НОУ**(x, t)
3. **Если** $\mathrm{x} \in \mathrm{Var}_t$ **и** $\mathrm{x} \neq t$**, то У**$(\mathrm{x}, t) = \emptyset$

Доказательство

2б) $\mathrm{x} \notin \mathrm{Var}_t$ и $\mathrm{x}\theta = t\theta \overset{?}{\Rightarrow} \exists\eta \quad \theta = \{\mathrm{x}/t\}\eta$

Достаточно показать, что $\theta = \{\mathrm{x}/t\}\theta$

Для этого рассмотрим произвольную переменную y и покажем, что $\mathrm{y}\theta = \mathrm{y}\{\mathrm{x}/t\}\theta$:

- Если $\mathrm{y} = \mathrm{x}$, то $\mathrm{y}\theta = \mathrm{x}\theta = t\theta = \mathrm{x}\{\mathrm{x}/t\}\theta = \mathrm{y}\{\mathrm{x}/t\}\theta$
- Иначе $\mathrm{y} \neq \mathrm{x}$, и $\mathrm{y}\theta = \mathrm{y}\{\mathrm{x}/t\}\theta$

Следовательно, для любой переменной y верно равенство $\mathrm{y}\theta = \mathrm{y}\{\mathrm{x}/t\}\theta$, а значит, верно и $\theta = \{\mathrm{x}/t\}\theta$

Унификация переменной и терма

Лемма (о связке). **Для любых переменной** x **и терма** t **верно:**

1. **Если** $x = t$**, то** $\varepsilon \in$ **НОУ**(x, t)
2. **Если** $\mathrm{x} \notin \mathrm{Var}_t$**, то** $\{\mathrm{x}/t\} \in$ **НОУ**(x, t)
3. **Если** $\mathrm{x} \in \mathrm{Var}_t$ **и** $\mathrm{x} \neq t$**, то У**$(\mathrm{x}, t) = \emptyset$

Доказательство

3. Рассмотрим произвольную подстановку θ и покажем,
что если $\mathrm{x} \in \mathrm{Var}_t$ и $\mathrm{x} \neq t$,
то θ не может являться унификатором x и t

Пусть $\mathrm{x}\theta = s$

Тогда $|\mathrm{x}\theta| = |s| < |t\{\mathrm{x}/s\}| \leq |t\theta|$ ($|p|$ — *длина терма* p)

Следовательно, $|\mathrm{x}\theta| < |t\theta|$, а значит, $\mathrm{x}\theta \neq t\theta$ ▼

Унификация приведённой системы

Система уравнений является приведённой, если она имеет вид

$$\begin{cases} \mathtt{x}_1 = t_1 \\ \quad \ldots \\ \mathtt{x}_k = t_k \end{cases},$$

где $\mathtt{x}_1, \ldots, \mathtt{x}_k$ — попарно различные переменные,
не встречающиеся в правых частях уравнений

Пример

$$\begin{cases} \mathtt{x} = \mathbf{f}(\mathtt{y}, \mathbf{g}(\mathtt{y})) \\ \mathtt{z} = \mathtt{w} \\ \mathtt{u} = \mathbf{g}(\mathbf{c}) \end{cases}$$ — **приведённая** система

Унификация приведённой системы

Система уравнений является приведённой, если она имеет вид

$$\begin{cases} \mathtt{x}_1 = t_1 \\ \ldots \\ \mathtt{x}_k = t_k \end{cases},$$

где $\mathtt{x}_1, \ldots, \mathtt{x}_k$ — попарно различные переменные,
не встречающиеся в правых частях уравнений

Пример

$$\begin{cases} \mathtt{x} = \mathbf{f}(\mathtt{y}, \mathbf{g}(\mathtt{y})) \\ \mathtt{x} = \mathtt{w} \\ \mathtt{y} = \mathbf{g}(\mathbf{c}, \mathbf{c}) \\ \mathbf{g}(\mathtt{z}) = \mathbf{f}(\mathbf{c}, \mathtt{x}) \end{cases}$$

— **неприведённая** система:

1. $\mathbf{g}(\mathtt{z})$ стоит в левой части уравнения, и это не переменная
2. $\mathtt{x}$ встречается в левых частях два раза
3. $\mathtt{y}$ встречается и в левой, и в правой частях

Унификация приведённой системы

Система уравнений является приведённой, если она имеет вид

$$\begin{cases} \mathtt{x}_1 = t_1 \\ \ldots \\ \mathtt{x}_k = t_k \end{cases},$$

где $\mathtt{x}_1, \ldots, \mathtt{x}_k$ — попарно различные переменные, не встречающиеся в правых частях уравнений

Лемма (о приведённой системе)

Если $\mathcal{E} = \begin{cases} \mathtt{x}_1 = t_1 \\ \ldots \\ \mathtt{x}_k = t_k \end{cases}$ **— приведённая система,**

то $\{\mathtt{x}_1/t_1, \ldots, \mathtt{x}_k/t_k\} \in$ **НОУ**$(\mathcal{E})$

Доказательство. Следует из леммы о связке ▼

Унификация произвольной системы

Алгоритм, о котором будет дальше идти речь, коротко описывается так:
это метод исключения переменных, широко применяющийся
для решения систем линейных алгебраических уравнений
и адаптированный к свободной алгебре термов

Напоминание, как работает этот метод
для уравнений над действительными числами:

$$\begin{cases} x = 3 - z + y \\ 2y + 3z = x + 2 \end{cases} \mapsto \begin{cases} x = 3 - z + y \\ 2y + 3z = 3 - z + y + 2 \end{cases} \longmapsto \begin{cases} x = 3 - z + y \\ y = 5 - 4z \end{cases}$$

$$\downarrow$$

$$\text{решения очевидны} \dashrightarrow \begin{cases} x = 8 - 5z \\ y = 5 - 4z \end{cases} \longmapsto\!\shortmid \begin{cases} x = 3 - z + 5 - 4z \\ y = 5 - 4z \end{cases}$$

В свободной алгебре термов вместо спектра арифметических операций
содержится только одна операция композиции,
но это не мешает адаптировать метод без особых усилий

Унификация произвольной системы

Алгоритм унификации ($\mathfrak{A}$):[1]

Далее будут описаны 6 правил преобразования системы уравнений

Эти правила **произвольно** (**недетерминированно**) применяются к системе, пока не станет верным одно из условий:

- получена приведённая система уравнений
  - ответ: унификатор из леммы о приведённой системе
- **явно** установлена невозможность унификации системы
  - ответ: СТОП: система неунифицируема

1 Martelli A., Montanari U. An efficient unification algorithm. 1982

Унификация произвольной системы

Правила преобразования системы уравнений

Упрощение системы: $(\mathtt{x} \in \mathrm{Var},\ t \in \mathrm{Term})$

Triv: удалить $t \doteq t$

Swap: заменить $t \doteq \mathtt{x}$ на $\mathtt{x} \doteq t$, если $t \notin \mathrm{Var}$

Func: заменить $\mathbf{f}(t_1, \ldots, t_k) \doteq \mathbf{f}(s_1, \ldots, s_k)$ на $\begin{cases} t_1 \doteq s_1 \\ \ldots \\ t_k \doteq s_k \end{cases}$

Elim: если в системе содержится уравнение $Eq : \mathtt{x} \doteq t$, где

- $\mathtt{x} \notin \mathrm{Var}_t$ и
- $\mathtt{x}$ встречается в других уравнениях системы

то применить подстановку $\{\mathtt{x}/t\}$ ко всем уравнениям системы, кроме Eq

Унификация произвольной системы

Правила преобразования системы уравнений

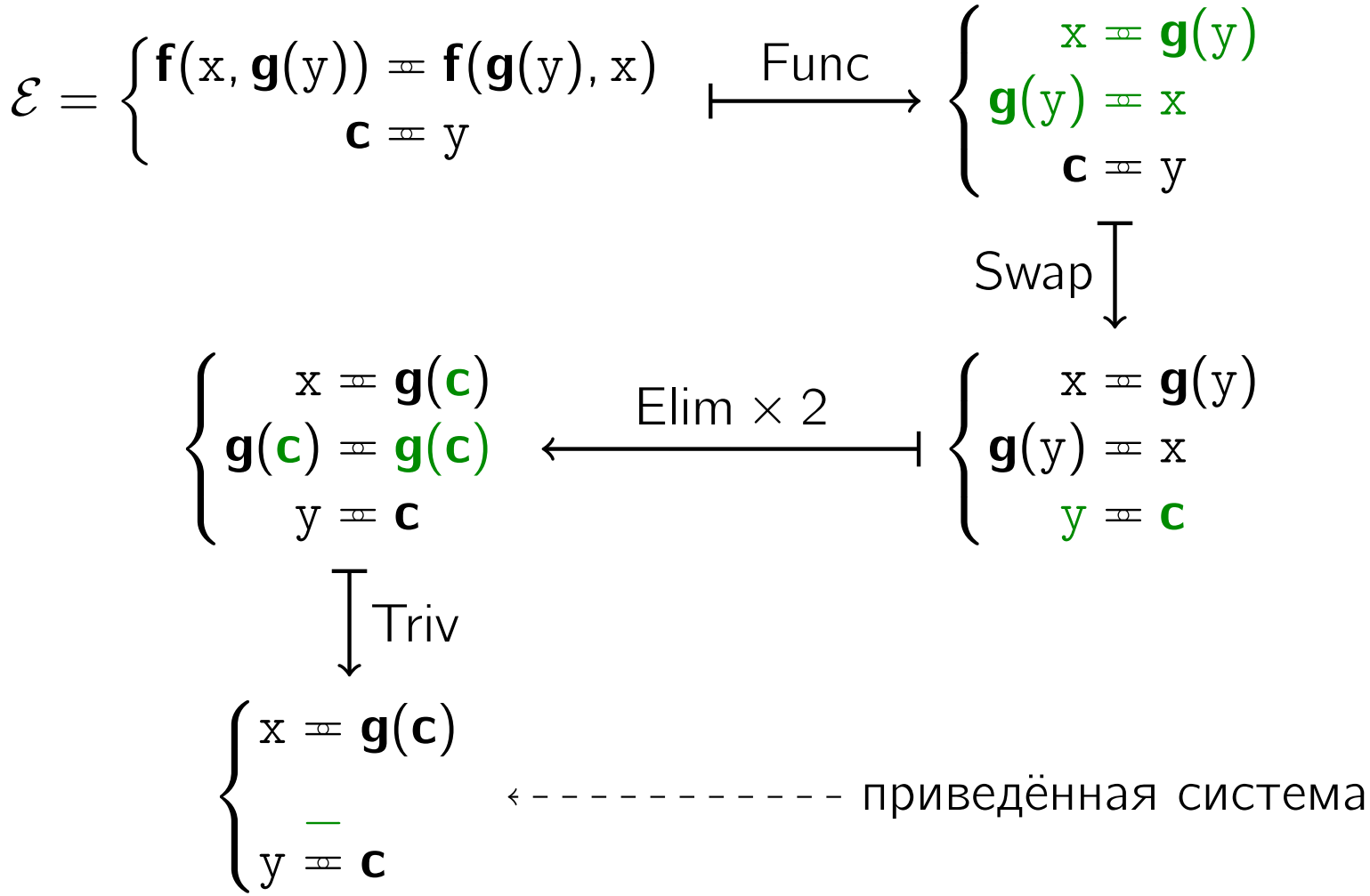
Явная неунифицируемость: $(\mathtt{x} \in \mathrm{Var},\ t \in \mathrm{Term})$

NElim: если в системе содержится уравнение $\mathtt{x} \eqdot t$,
где $\mathtt{x} \in \mathrm{Var}_t$ и $\mathtt{x} \neq t$, то
СТОП: система неунифицируема

NFunc: если в системе содержится уравнение
$\mathbf{f}(t_1, \ldots, t_k) \eqdot \mathbf{g}(s_1, \ldots, s_m)$, где $\mathbf{f} \neq \mathbf{g}$, то
СТОП: система неунифицируема

Унификация произвольной системы

Примеры



Ответ: $\{\mathrm{x}/\mathbf{g}(\mathbf{c}),\ \mathrm{y}/\mathbf{c}\} \in \mathrm{НОУ}(\mathcal{E})$

Унификация произвольной системы

Примеры

$$\mathcal{E}' = \begin{cases} \mathbf{f}(\mathrm{x}, \mathbf{g}(\mathrm{x})) \eqdot \mathbf{h}(\mathbf{g}(\mathrm{y}), \mathrm{x}) \\ \mathbf{c} \eqdot \mathrm{y} \end{cases} \xmapsto{\text{NFunc}} \text{СТОП}$$

Ответ: $\mathsf{У}(\mathcal{E}') = \emptyset$

$$\mathcal{E}'' = \begin{cases} \mathbf{f}(\mathrm{x}, \mathbf{g}(\mathrm{x})) \eqdot \mathbf{f}(\mathbf{g}(\mathrm{y}), \mathrm{x}) \\ \mathbf{c} \eqdot \mathrm{y} \end{cases} \xmapsto{\text{Func}} \begin{cases} \mathrm{x} \eqdot \mathbf{g}(\mathrm{y}) \\ \mathbf{g}(\mathrm{x}) \eqdot \mathrm{x} \\ \mathbf{c} \eqdot \mathrm{y} \end{cases}$$

$$\downarrow \text{Swap}$$

$$\text{СТОП} \xleftarrow{\text{NElim}} \begin{cases} \mathrm{x} \eqdot \mathbf{g}(\mathrm{y}) \\ \mathrm{x} \eqdot \mathbf{g}(\mathrm{x}) \\ \mathbf{c} \eqdot \mathrm{y} \end{cases}$$

Ответ: $\mathsf{У}(\mathcal{E}'') = \emptyset$

Унификация произвольной системы

Теорема (об унификации). **Для любой системы уравнений $\mathcal{E}_0$**

- **алгоритм $\mathfrak{A}$ завершает работу на $\mathcal{E}_0$** **(завершаемость)**
- **по завершении алгоритмом $\mathfrak{A}$ выдаётся подстановка или сообщение СТОП** **(успешность)**
- **если выдана подстановка θ, то $\theta \in$ НОУ$(\mathcal{E}_0)$** **(корректность)**
- **если выдано сообщение СТОП, то система $\mathcal{E}$ неунифицируема** **(полнота)**

Следствие. **Для любых атомов A и B логики предикатов верно:**

$$\text{У}(A, B) \neq \emptyset \quad \Leftrightarrow \quad \text{НОУ}(A, B) \neq \emptyset$$

Доказательство теоремы об унификации

Завершаемость ($\mathcal{E}_0 \not\mapsto \infty$)

Далее будет строго сформулированы и обоснованы следующие факты:

- на каждом шаге работы алгоритма система становится проще
- после конечного числа шагов работы алгоритма обязательно получается система, которую нельзя упростить

Предложим характеристику системы, которая
убывает на каждом шаге и при этом
не может убывать бесконечно долго

Завершаемость алгоритма
очевидным образом следует из существования такой характеристики:

если система упрощается на каждом шаге
и не может упрощаться бесконечно,
то число шагов конечно

Доказательство теоремы об унификации

Завершаемость $(\mathcal{E}_0 \not\leadsto \infty)$

Переменную $\mathtt{x}$ назовём приведённой в системе $\mathcal{E}$, если $\mathcal{E}$

- содержит уравнение вида $\mathtt{x} = t$, где $\mathtt{x} \notin \mathrm{Var}_t$, и
- не содержит $\mathtt{x}$ в других уравнениях

Характеристикой системы $\mathcal{E}$ объявим упорядоченную тройку чисел $\mathcal{H}(\mathcal{E}) = \langle vr(\mathcal{E}), fs(\mathcal{E}), eq(\mathcal{E})\rangle$, где

- $vr(\mathcal{E})$ — число неприведённых переменных системы $\mathcal{E}$
- $fs(\mathcal{E})$ — суммарное число функциональных символов и констант в левых частях уравнений $\mathcal{E}$
- $eq(\mathcal{E})$ — число уравнений системы $\mathcal{E}$

Лексикографический порядок на тройках целых чисел определяется так:

$$\langle n_1, n_2, n_3\rangle \succ \langle m_1, m_2, m_3\rangle \Leftrightarrow \left[\begin{array}{l} n_1 > m_1 \\ n_1 = m_1,\ \ n_2 > m_2 \\ n_1 = m_1,\ \ n_2 = m_2,\ \ n_3 > m_3 \end{array}\right.$$

Пример: $\langle 2, 11, 2\rangle \succ \langle 2, 10, 5578\rangle \succ \langle 2, 10, 5577\rangle \succ \langle 1, 1001, 78\rangle$

Доказательство теоремы об унификации

Завершаемость $(\mathcal{E}_0 \not\mapsto \infty)$

$\mathcal{H}(\mathcal{E}) = \langle vr(\mathcal{E}), fs(\mathcal{E}), eq(\mathcal{E})\rangle$ $vr(\mathcal{E})$: число неприведённых переменных

$fs(\mathcal{E})$: число функциональных символов и констант в левых частях $eq(\mathcal{E})$: число уравнений

Факт 1: при применении правил упрощения характеристика системы уменьшается относительно $\succ$

$$\mathcal{E}_1 = \left\{ \begin{array}{l} \ldots \\ t \eqdot t \\ \ldots \end{array} \right. \quad \stackrel{\text{Triv}}{\longmapsto} \quad \mathcal{E}_1' = \left\{ \begin{array}{l} \ldots \\ \\ \ldots \end{array} \right.$$

$$vr(\mathcal{E}_1) \geq vr(\mathcal{E}_1') \qquad fs(\mathcal{E}_1) \geq fs(\mathcal{E}_1') \qquad eq(\mathcal{E}_1) > eq(\mathcal{E}_1')$$

$$\mathcal{E}_2 = \left\{ \begin{array}{l} \ldots \\ t \eqdot \mathrm{x} \\ \ldots \end{array} \right. \quad \stackrel{\text{Swap}}{\longmapsto} \quad \mathcal{E}_2' = \left\{ \begin{array}{l} \ldots \\ \mathrm{x} \eqdot t \\ \ldots \end{array} \right.$$

$$vr(\mathcal{E}_2) \geq vr(\mathcal{E}_2') \qquad fs(\mathcal{E}_2) > fs(\mathcal{E}_2')$$

Доказательство теоремы об унификации

Завершаемость $(\mathcal{E}_0 \not\mapsto \infty)$

$\mathcal{H}(\mathcal{E}) = \langle vr(\mathcal{E}), fs(\mathcal{E}), eq(\mathcal{E})\rangle$ $vr(\mathcal{E})$: число неприведённых переменных

$fs(\mathcal{E})$: число функциональных символов и констант в левых частях $eq(\mathcal{E})$: число уравнений

Факт 1: при применении правил упрощения характеристика системы уменьшается относительно $\succ$

$$\mathcal{E}_3 = \left\{ \begin{array}{l} \ldots \\ \mathbf{f}(t_1, \ldots, t_k) \doteqdot \mathbf{f}(s_1, \ldots, s_k) \\ \ldots \end{array} \right. \xrightarrow{\text{Func}} \mathcal{E}'_3 = \left\{ \begin{array}{l} \ldots \\ t_1 \doteqdot s_1 \\ \ldots \\ t_k \doteqdot s_k \\ \ldots \end{array} \right.$$

$$vr(\mathcal{E}_3) \geq vr(\mathcal{E}'_3) \qquad fs(\mathcal{E}_3) > fs(\mathcal{E}'_3)$$

$$\mathcal{E}_4 = \left\{ \begin{array}{l} \ldots \\ \mathrm{x} \doteqdot t \\ \ldots \end{array} \right. \xrightarrow{\text{Elim}} \mathcal{E}'_4 = \left\{ \begin{array}{l} \ldots\{\mathrm{x}/t\} \\ \mathrm{x} \doteqdot t \\ \ldots\{\mathrm{x}/t\} \end{array} \right.$$

$$vr(\mathcal{E}_4) > vr(\mathcal{E}'_4)$$

Доказательство теоремы об унификации

Завершаемость ($\mathcal{E}_0 \not\mapsto \infty$)

$\mathcal{H}(\mathcal{E}) = \langle vr(\mathcal{E}), fs(\mathcal{E}), eq(\mathcal{E})\rangle$ $vr(\mathcal{E})$: число неприведённых переменных
$fs(\mathcal{E})$: число функциональных символов и констант в левых частях $eq(\mathcal{E})$: число уравнений

Факт 2: характеристика не может бесконечно долго убывать относительно $\succ$

Чтобы «прочувствовать» этот факт, представьте, что характеристика $\langle a, b, c\rangle$ отвечает мешку конфет: a вкусных, b обычных и c невкусных

Тогда убывание характеристики можно представить как одну из операций в пункте обмена конфет:

- отдать вкусную конфету
  и взамен получить сколько угодно обычных и невкусных
- отдать обычную конфету
  и взамен получить сколько угодно невкусных
- отдать невкусную конфету

Оказывается, что так меняться конфетами невыгодно,
и если слишком увлечься, то мешок непременно опустеет

Доказательство теоремы об унификации

Завершаемость ($\mathcal{E}_0 \not\mapsto \infty$)

$\mathcal{H}(\mathcal{E}) = \langle vr(\mathcal{E}), fs(\mathcal{E}), eq(\mathcal{E})\rangle$ $vr(\mathcal{E})$: число неприведённых переменных

$fs(\mathcal{E})$: число функциональных символов и констант в левых частях $eq(\mathcal{E})$: число уравнений

Факт 2: характеристика не может бесконечно долго убывать относительно $\succ$

Лемма. Не существует бесконечной последовательности троек неотрицательных целых чисел, убывающей относительно $\succ$

Доказательство опустим, так как это уже не совсем логика

Отсутствие бесконечных убывающих последовательностей элементов в частично упорядоченном множестве (ЧУМ) принято называть свойством обрыва убывающих цепей и фундированностью множества

Фундированные ЧУМ нередко используются так же, как здесь, для обоснования завершаемости алгоритмов

Доказательство теоремы об унификации

Успешность ($\mathcal{E}_0 \rightsquigarrow \theta$/СТОП)

Предположим, что алгоритм неуспешен

Это означает, что на очередном шаге получена неприведённая система $\mathcal{E}$, к которой невозможно применить ни одно из правил Triv, Swap, Func, Elim, NFunc, NElim

Невозможно применить правила Triv, Swap, Func, NFunc $\Rightarrow$
в левых частях $\mathcal{E}$ содержатся **только** переменные

Невозможно применить правила Elim, NElim $\Rightarrow$
все переменные в левых частях $\mathcal{E}$ являются приведёнными

Следовательно, $\mathcal{E}$ — система, в левых частях которой располагаются **только** приведённые переменные,
то есть $\mathcal{E}$ — приведённая система (*противоречие*)

Доказательство теоремы об унификации

Корректность $(\mathcal{E}_0 \rightsquigarrow \theta \Rightarrow \theta \in \text{НОУ}(\mathcal{E}_0))$

Системы уравнений $\mathcal{E}$ и $\mathcal{E}'$ равносильны, если $\text{У}(\mathcal{E}) = \text{У}(\mathcal{E}')$

Достаточно показать, что при применении правил упрощения
(Triv, Swap, Func, Elim)
обязательно получается система, равносильная исходной

Для правил Triv, Swap и Func это довольно просто,
так что подробно остановимся только на «трудном» правиле Elim

$$\mathcal{E} = \begin{cases} \ldots \\ \mathrm{x} = t \\ \ldots \end{cases} \xrightarrow{\text{Elim}} \mathcal{E}' = \begin{cases} \ldots \{\mathrm{x}/t\} \\ \mathrm{x} = t \\ \ldots \{\mathrm{x}/t\} \end{cases}$$

Покажем, что системы $\mathcal{E}$ и $\mathcal{E}'$ равносильны

Доказательство теоремы об унификации

Корректность $(\mathcal{E}_0 \rightsquigarrow \theta \Rightarrow \theta \in \text{НОУ}(\mathcal{E}_0))$

$$\mathcal{E} = \begin{cases} \dots \\ \mathrm{x} = t \\ \dots \end{cases} \xrightarrow{\text{Elim}} \mathcal{E}' = \begin{cases} \dots\{\mathrm{x}/t\} \\ \mathrm{x} = t \\ \dots\{\mathrm{x}/t\} \end{cases}$$

$$\text{У}(\mathcal{E}) \stackrel{?}{=} \text{У}(\mathcal{E}')$$

($\subseteq$): Пусть $\eta \in \text{У}(\mathcal{E})$

Тогда $\mathrm{x}\eta = t\eta$

Из доказательства леммы о связке: $\eta = \{\mathrm{x}/t\}\eta$, а значит,

$$\begin{cases} \dots\eta \\ \mathrm{x}\eta = t\eta \\ \dots\eta \end{cases} \quad \Leftrightarrow \quad \begin{cases} \dots\{\mathrm{x}/t\}\eta \\ \mathrm{x}\eta = t\eta \\ \dots\{\mathrm{x}/t\}\eta \end{cases}$$

Следовательно, η — унификатор системы $\mathcal{E}'$

($\supseteq$): Рассуждения аналогичны

Доказательство теоремы об унификации

Полнота ($\mathcal{E}_0 \rightsquigarrow$ СТОП $\Rightarrow$ У$(\mathcal{E}_0) = \emptyset$)

Для определённости положим, что «СТОП» выдано для системы $\mathcal{E}$

Пусть для этого было применено правило NFunc

Тогда $\mathcal{E}$ содержит уравнение $\mathbf{f}(\dots) \eqdot \mathbf{g}(\dots)$, где $\mathbf{f} \neq \mathbf{g}$

Ни для какой подстановки θ не верно $\mathbf{f}(\dots)\theta = \mathbf{g}(\dots)\theta$

Иначе для этого было применено правило NElim

Тогда $\mathcal{E}$ содержит уравнение $\mathrm{x} \eqdot t$, где $\mathrm{x} \in \mathrm{Var}_t$ и $\mathrm{x} \neq t$

По лемме о связке, ни для какой подстановки θ не верно $\mathrm{x}\theta = t\theta$

Следовательно, У$(\mathcal{E}) = \emptyset$

Система $\mathcal{E}$ была получена из $\mathcal{E}_0$ применением правил

Triv, Swap, Func, Elim

Согласно рассуждениям в обосновании корректности алгоритма, системы $\mathcal{E}$ и $\mathcal{E}_0$ равносильны

Значит, У$(\mathcal{E}_0) = \emptyset$ ▼

Математическая логика и логическое программирование

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

Блок 21

Монотонность и транзитивность отношения логического следования

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

ВМК МГУ, 2023/2024, осенний семестр

Монотонность следования в ЛП

Для любых множеств предложений Γ, Δ и любого предложения φ верно:

$$\Gamma \models \varphi \quad \Rightarrow \quad \Gamma \cup \Delta \models \varphi$$

Доказательство

Пусть верно $\Gamma \models \varphi$

Тогда для любой интерпретации $\mathcal{I}$ верна цепочка следований:

$\mathcal{I} \models \Gamma \cup \Delta$

$\Rightarrow$ (по определению модели)

для любой формулы ψ из $\Gamma \cup \Delta$ верно $\mathcal{I} \models \psi$

$\Rightarrow$ (т.к. $\Gamma \subseteq \Gamma \cup \Delta$)

для любой формулы ψ из Γ верно $\mathcal{I} \models \psi$

$\Rightarrow$ (по определению модели)

$\mathcal{I} \models \Gamma$

$\Rightarrow$ (по определению логического следствия и т.к. $\Gamma \models \varphi$)

$\mathcal{I} \models \varphi$

Значит, любая модель $\Gamma \cup \Delta$ является моделью φ, то есть $\Gamma \cup \Delta \models \varphi$ ▼

Монотонность $\vee$ относительно следования в ЛП

Для любого множества предложений Γ
и любых формул $\varphi(\widetilde{x}^n)$, $\psi(\widetilde{x}^n)$ верно:

$$\Gamma \models \forall \widetilde{x}^n \, \varphi \quad \Rightarrow \quad \Gamma \models \forall \widetilde{x}^n \, (\varphi \vee \psi)$$

Доказательство

Пусть $\Gamma \models \forall \widetilde{x}^n \, \varphi$

Тогда для любой интерпретации $\mathcal{I}$ верна цепочка следований:

$\mathcal{I} \models \Gamma$

$\Rightarrow$ (по определению логического следствия и т.к. $\Gamma \models \forall \widetilde{x}^n \, \varphi$)

$\mathcal{I} \models \forall \widetilde{x}^n \, \varphi$

$\Rightarrow$ (по семантике $\forall$)

для любого набора предметов $\widetilde{d}^n$ верно $\mathcal{I} \models \varphi[\widetilde{d}^n]$

$\Rightarrow$ (по семантике $\vee$)

для любого набора предметов $\widetilde{d}^n$ верно $\mathcal{I} \models (\varphi \vee \psi)[\widetilde{d}^n]$

$\Rightarrow$ (по семантике $\forall$)

$\mathcal{I} \models \forall \widetilde{x}^n \, (\varphi \vee \psi)$

Значит, любая модель Γ является моделью $\forall \widetilde{x}^n \, (\varphi \vee \psi)$ ▼

Транзитивность следования в ЛП

Для любого множества предложений Γ и любых предложений $\psi_1, \ldots, \psi_k, \varphi$ верно:

$$\left.\begin{array}{l} \Gamma \models \psi_1 \\ \ldots \\ \Gamma \models \psi_k \\ \psi_1, \ldots, \psi_k \models \varphi \end{array}\right\} \quad \Rightarrow \quad \Gamma \models \varphi$$

Доказательство

Пусть верны все соотношения слева от «$\Rightarrow$»

Тогда для любой интерпретации $\mathcal{I}$ верна цепочка следований:

$\mathcal{I} \models \Gamma$

$\Rightarrow$ (по определению логического следования)

$\mathcal{I} \models \psi_1$ и $\ldots$ и $\mathcal{I} \models \psi_k$

$\Rightarrow$ (по определению модели множества формул)

$\mathcal{I} \models \{\psi_1, \ldots, \psi_k\}$

$\Rightarrow$ (по определению логического следования)

$\mathcal{I} \models \varphi$ ▼

Математическая логика и логическое программирование

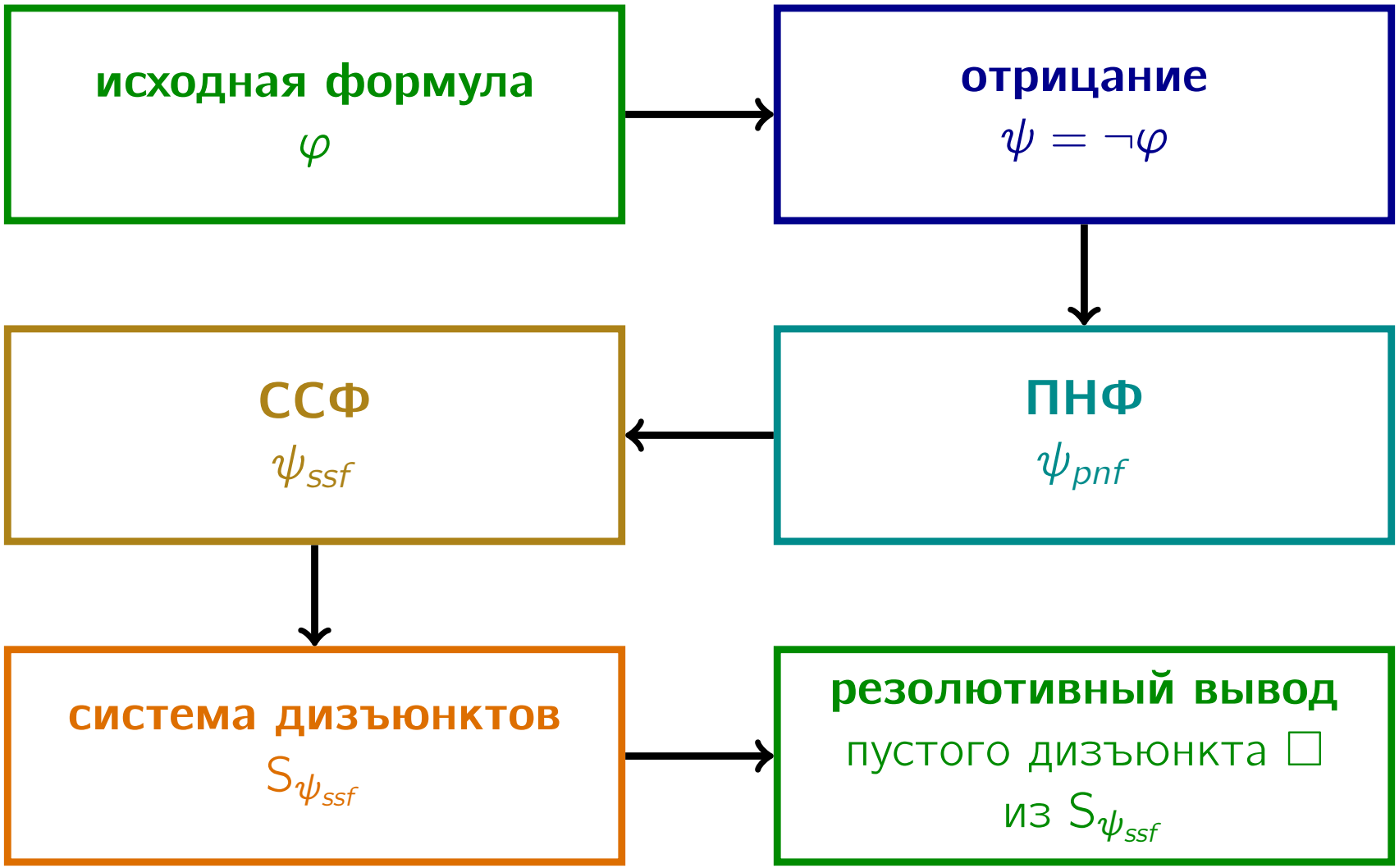
mk.cs.msu.ru → Лекционные курсы
→ Математическая логика и логическое программирование (3-й поток)

Блок 22

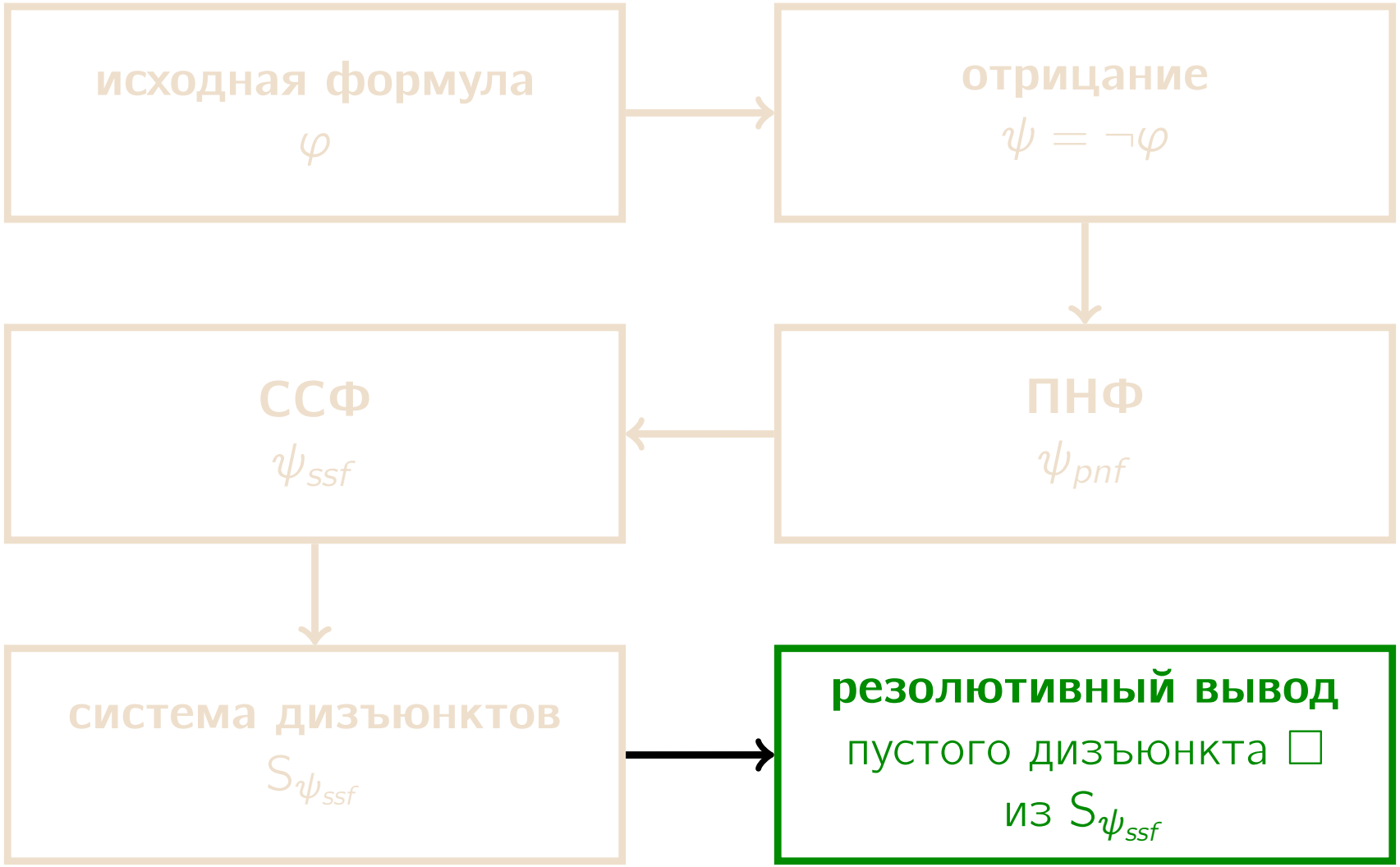
Резолютивный вывод
Корректность резолютивного вывода

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

ВМК МГУ, 2023/2024, осенний семестр



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf} \quad \Leftrightarrow \quad \not\models S_{\psi_{ssf}}$$



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf} \quad \Leftrightarrow \quad \not\models S_{\psi_{ssf}}$$

Ещё немного определений

Положительная литера — это атом

Отрицательная литера — это отрицание атома

Если E — логическое выражение и θ — подстановка, то:

- $E\theta$ — пример выражения E
- если $\mathrm{Var}_{E\theta} = \emptyset$, то $E\theta$ — основной пример
- если $\theta : \mathrm{Var} \to \mathrm{Var}$ — биекция, то
  - θ — переименование
  - $E\theta$ — вариант выражения E

Ещё немного определений

Пример

Рассмотрим выражение $E = \mathrm{P}(\mathrm{x}, \mathbf{f}(\mathrm{y})) \vee \neg \mathrm{R}(\mathrm{y}, \mathbf{c})$ и подстановки

$$\theta = \{\mathrm{x}/\mathrm{u}, \mathrm{y}/\mathrm{z}, \mathrm{u}/\mathrm{x}, \mathrm{z}/\mathrm{y}\}$$
$$\eta = \{\mathrm{x}/\mathbf{g}(\mathbf{d}), \mathrm{y}/\mathrm{z}\}$$
$$\mu = \{\mathrm{z}/\mathbf{c}\}$$
$$\varepsilon = \{\}$$

Тогда:

- подстановки θ и ε — переименования, а η и μ — нет
- $E\eta = \mathrm{P}(\mathbf{g}(\mathbf{d}), \mathbf{f}(\mathrm{z})) \vee \neg \mathrm{R}(\mathrm{z}, \mathbf{c})$ — пример выражения E
- $E\eta\mu = \mathrm{P}(\mathbf{g}(\mathbf{d}), \mathbf{f}(\mathbf{c})) \vee \neg \mathrm{R}(\mathbf{c}, \mathbf{c})$ — основной пример выражения E
- $E\theta = \mathrm{P}(\mathrm{u}, \mathbf{f}(\mathrm{z})) \vee \neg \mathrm{R}(\mathrm{z}, \mathbf{c})$ — вариант выражения E

Правило резолюции

Правило резолюции:

$$\frac{D_1 \vee L_1,\ D_2 \vee \neg L_2}{(D_1 \vee D_2)\theta}$$

Здесь

- D_1, D_2 — дизъюнкты
- L_1, L_2 — положительные литеры
- $\theta \in \text{НОУ}(L_1, L_2)$

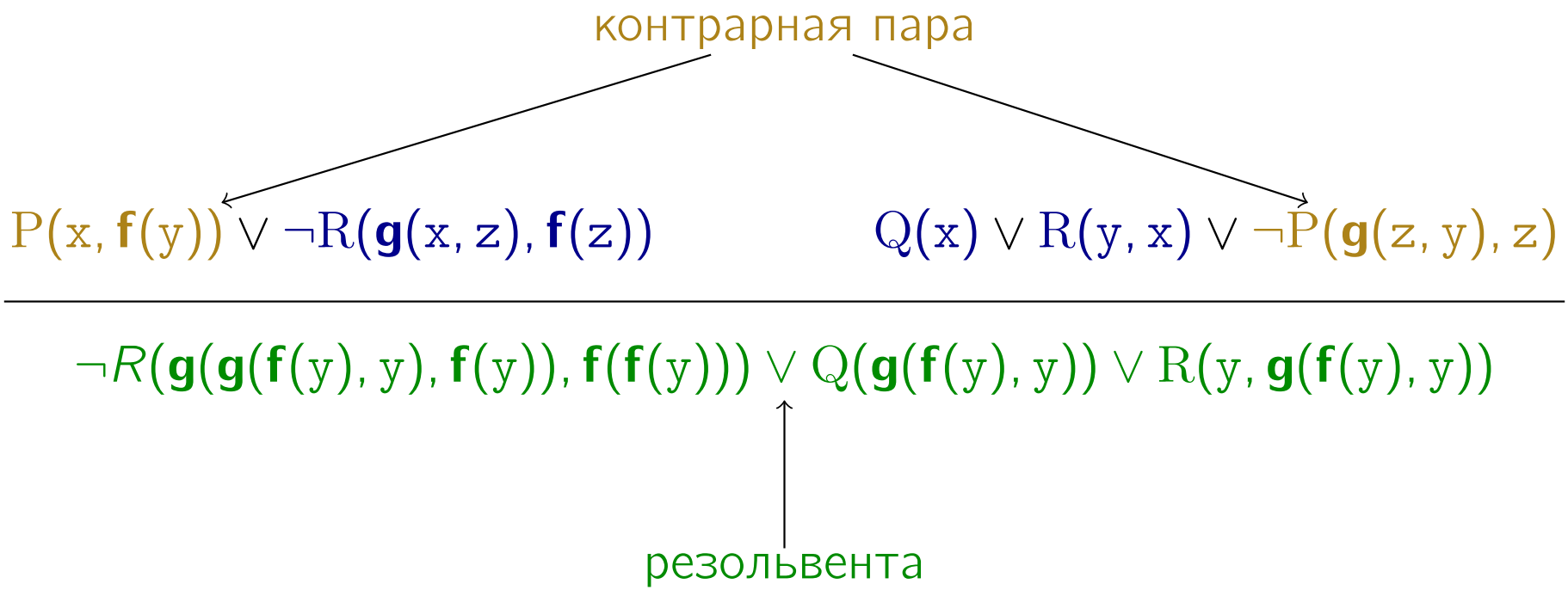
При использовании правила резолюции
допускается перестановка слагаемых дизъюнктов

Дизъюнкт $(D_1 \vee D_2)\theta$ — резольвента дизъюнктов $D_1 \vee L_1$, $D_2 \vee \neg L_2$

Литеры L_1, $\neg L_2$ образуют контрарную пару

Правило резолюции

Пример

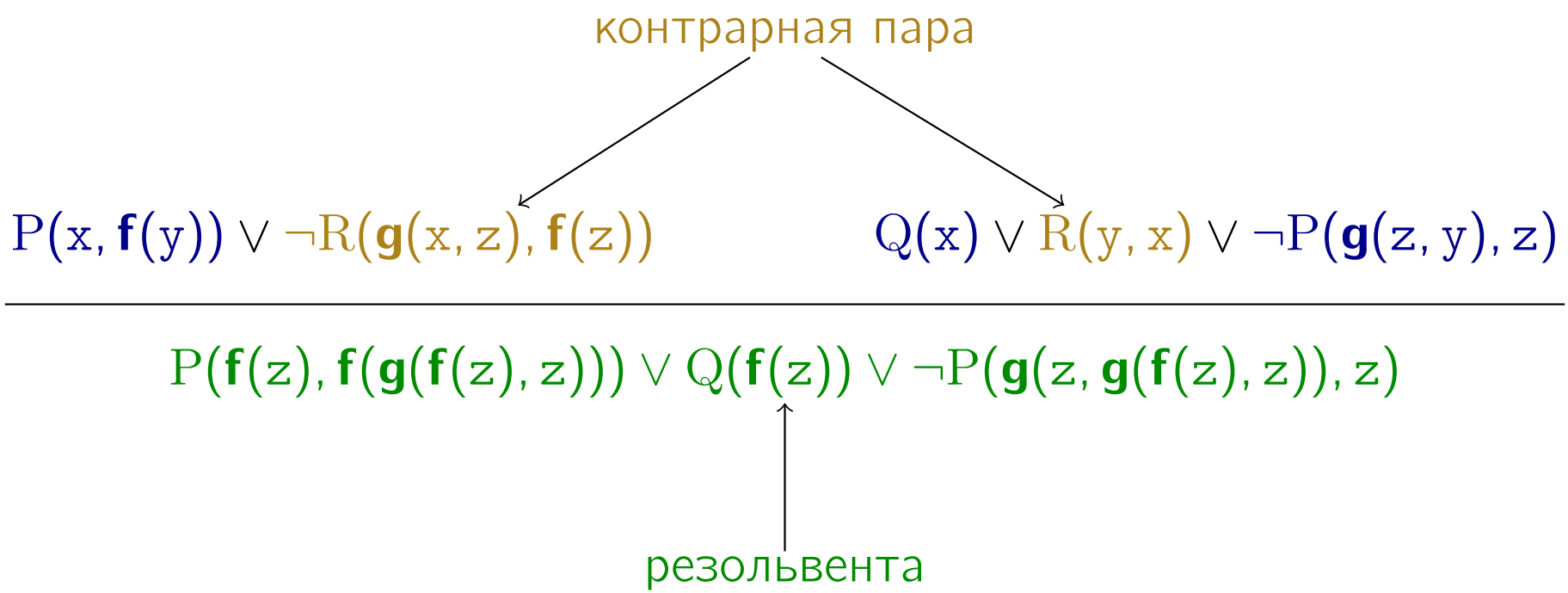


$$\theta = \{\mathrm{x}/\mathbf{g}(\mathbf{f}(\mathrm{y}), \mathrm{y}),\ \mathrm{z}/\mathbf{f}(\mathrm{y})\} \in \mathsf{НОУ}(\mathrm{P}(\mathrm{x}, \mathbf{f}(\mathrm{y})), \mathrm{P}(\mathbf{g}(\mathrm{z}, \mathrm{y}), \mathrm{z}))$$

$$\text{резольвента: } (\neg \mathrm{R}(\mathbf{g}(\mathrm{x}, \mathrm{z}), \mathbf{f}(\mathrm{z})) \vee \mathrm{Q}(\mathrm{x}) \vee \mathrm{R}(\mathrm{y}, \mathrm{x}))\theta$$

Правило резолюции

Пример

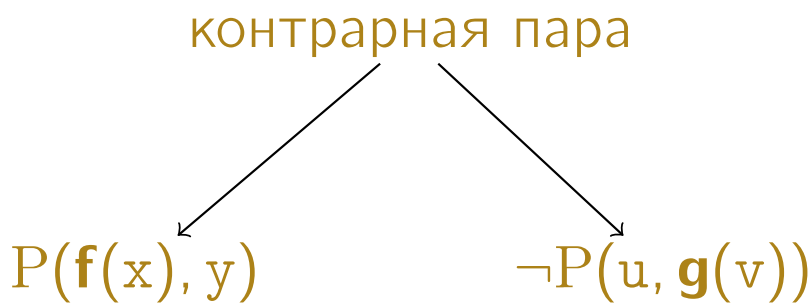


$$\theta = \{\mathrm{x}/\mathbf{f}(\mathrm{z}),\ \mathrm{y}/\mathbf{g}(\mathbf{f}(\mathrm{z}), \mathrm{z})\} \in \mathsf{НОУ}(\mathrm{R}(\mathbf{g}(\mathrm{x}, \mathrm{z}), \mathbf{f}(\mathrm{z})), \mathrm{R}(\mathrm{y}, \mathrm{x}))$$

$$\text{резольвента: } (\mathrm{P}(\mathrm{x}, \mathbf{f}(\mathrm{y})) \vee \mathrm{Q}(\mathrm{x}) \vee \neg \mathrm{P}(\mathbf{g}(\mathrm{z}, \mathrm{y}), \mathrm{z}))\theta$$

Правило резолюции

Пример

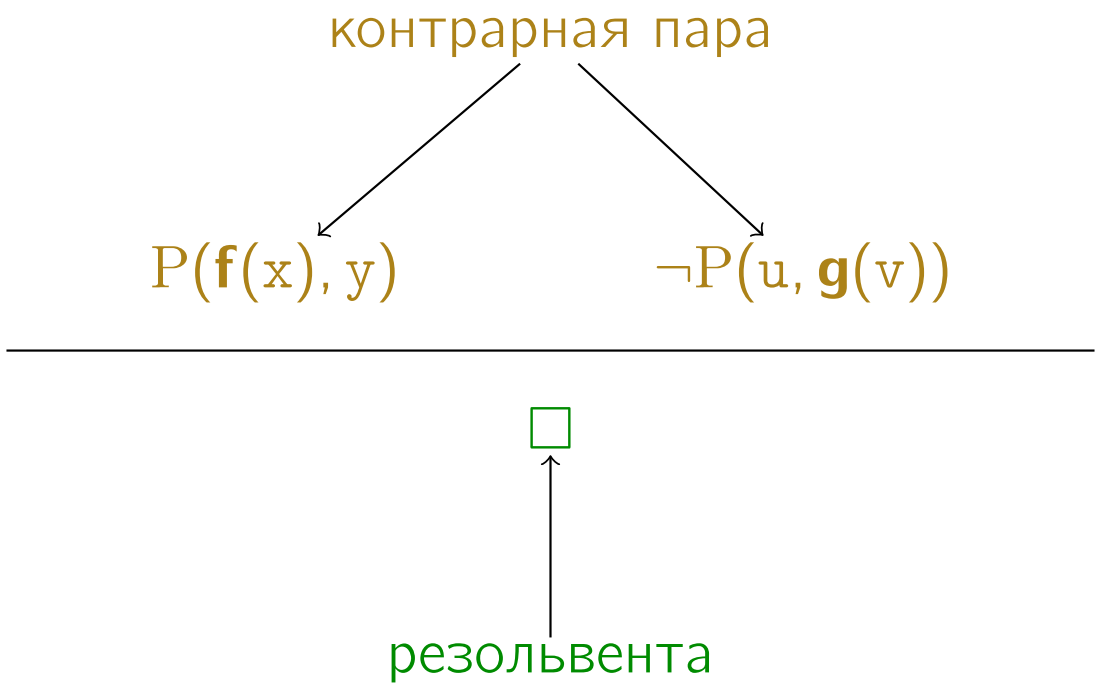


$$\theta = \{\mathrm{y}/\mathbf{g}(\mathrm{v}), \mathrm{u}/\mathbf{f}(\mathrm{x})\} \in \text{НОУ}(\mathrm{P}(\mathbf{f}(\mathrm{x}), \mathrm{y}), \mathrm{P}(\mathrm{u}, \mathbf{g}(\mathrm{v}))$$

резольвента: $(???)\theta$

Правило резолюции

Пример



$$\theta = \{\mathrm{y}/\mathbf{g}(\mathrm{v}), \mathrm{u}/\mathbf{f}(\mathrm{x})\} \in \text{НОУ}(\mathrm{P}(\mathbf{f}(\mathrm{x}), \mathrm{y}), \mathrm{P}(\mathrm{u}, \mathbf{g}(\mathrm{v}))$$

резольвента: (пустое мультимножество литер)θ

Правило резолюции

Лемма (о корректности правила резолюции)
Если D **— резольвента дизъюнктов** D_1**,** D_2**, то** $D_1, D_2 \models D$

Доказательство (кванторные приставки опущены)

Пусть $D_1 = D_1' \vee L_1$, $D_2 = D_2' \vee \neg L_2$, $\theta \in \text{НОУ}(L_1, L_2)$,
$D = (D_1' \vee D_2')\theta$ и $L_1\theta = L_2\theta = L$

Тогда по свойствам отношения $\models$ и операции $\vee$ верно следующее:

$D_1 \models D_1\theta$	$D_2 \models D_2\theta$
$D_1 \models D_1'\theta \vee L_1\theta$	$D_2 \models D_2'\theta \vee \neg L_2\theta$
$D_1 \models D_1'\theta \vee L$	$D_2 \models D_2'\theta \vee \neg L$
$D_1, D_2 \models D_1'\theta \vee L$	$D_1, D_2 \models D_2'\theta \vee \neg L$
$D_1, D_2 \models D_1'\theta \vee D_2'\theta \vee L$	$D_1, D_2 \models D_1'\theta \vee D_2'\theta \vee \neg L$

Заметим, что если $\Gamma \models A \vee B$ и $\Gamma \models A \vee \neg B$, то $\Gamma \models A$ (очевидно?)

Тогда $D_1, D_2 \models D_1'\theta \vee D_2'\theta$

То есть $D_1, D_2 \models D$ ▼

Правило склейки

Применение одного только правила резолюции далеко не всегда позволяет вывести $\square$ из невыполнимой системы

Например:

$$\mathrm{P}(\mathrm{x}) \vee \mathrm{P}(\mathbf{c})$$
$$\neg\mathrm{P}(\mathbf{c}) \vee \neg\mathrm{P}(\mathrm{y})$$

Система из таких двух дизъюнктов невыполнима, но все резольвенты, резольвенты резольвент и т.д.
этих дизъюнктов имеют ровно две литеры

Необходимо иметь правило,
которое позволит получать $\square$ и из таких систем

Правило склейки

Правило склейки

$$\frac{D \vee L_1 \vee L_2}{(D \vee L_1)\theta}$$

Здесь

- D — дизъюнкт
- L_1, L_2 — литеры
- $\theta \in \text{НОУ}(L_1, L_2)$

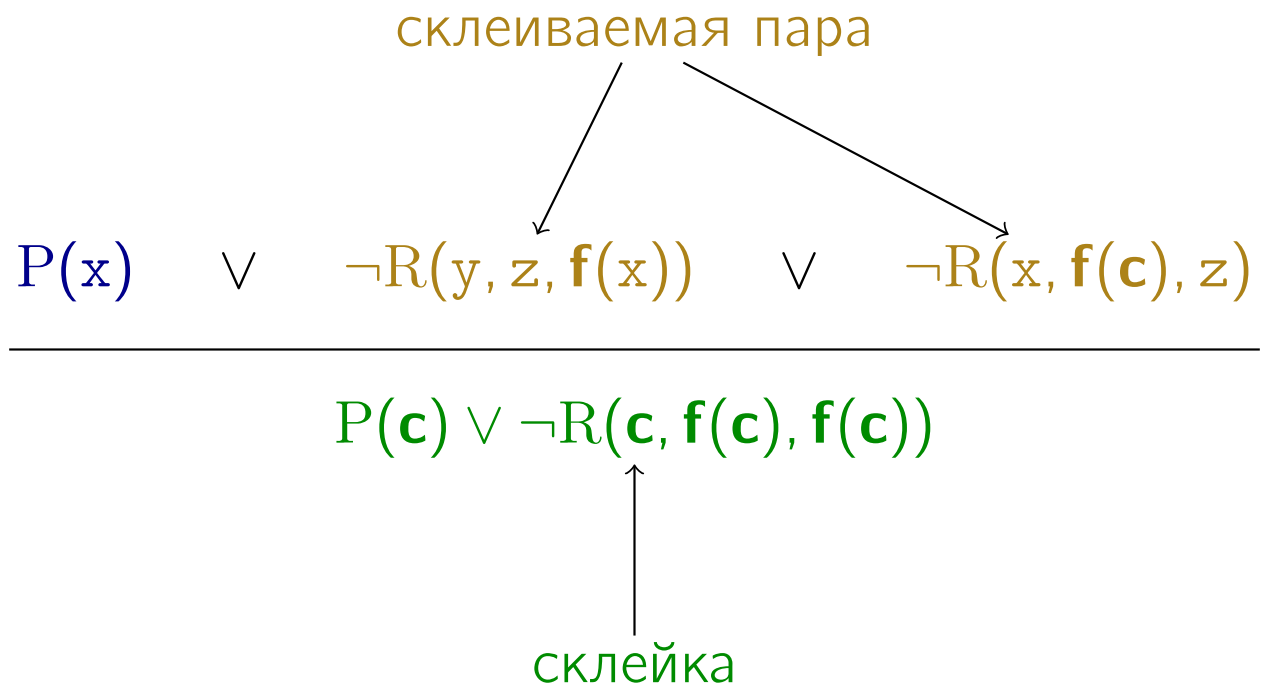
При использовании правила склейки
допускается перестановка слагаемых дизъюнктов

Дизъюнкт $(D \vee L_1)\theta$ — склейка дизъюнкта $D \vee L_1 \vee L_2$

Литеры L_1, L_2 образуют склеиваемую пару

Правило склейки

Пример



$$\theta = \{\mathrm{x}/\mathbf{c}, \mathrm{y}/\mathbf{c}, \mathrm{z}/\mathbf{f}(\mathbf{c})\} \in \text{НОУ}(\neg\mathrm{R}(\mathrm{y},\mathrm{z},\mathbf{f}(\mathrm{x})), \neg\mathrm{R}(\mathrm{x},\mathbf{f}(\mathbf{c}),\mathrm{z}))$$

склейка: $(\mathrm{P}(\mathrm{x}) \vee \neg\mathrm{R}(\mathrm{y},\mathrm{z},\mathbf{f}(\mathrm{x})))\theta$

Лемма (о корректности правила склейки)

Если D **— склейка дизъюнкта** D_1**, то** $\mathrm{D}_1 \models \mathrm{D}$

Доказательство. Аналогично доказательству леммы о корректности правила резолюции

Резолютивный вывод

Пусть S — система дизъюнктов

Резолютивный вывод из S — это
конечная последовательность дизъюнктов

$$\mathrm{D}_1,\ \ldots,\ \mathrm{D}_i,\ \ldots,\ \mathrm{D}_k,$$

такая что каждый дизъюнкт D_i является

- вариантом дизъюнкта из S,
- склейкой дизъюнкта D_j, где $j < i$, или
- резольвентой дизъюнктов D_j, D_m, где $j < i$ и $m < i$

Дизъюнкт резолютивно выводи́м из S, если существует
резолютивный вывод из S, оканчивающийся этим дизъюнктом

Резолютивный вывод

Пример

$$S = \left\{ \begin{array}{c} \mathrm{P}(\mathrm{x}) \\ \neg\mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x}, \mathbf{g}(\mathrm{x})) \\ \neg\mathrm{R}(\mathrm{x}, \mathrm{u}) \end{array} \right\}$$

Резолютивный вывод $\square$ из S:

вариант	$\mathrm{P}(\mathrm{x}_1)$
вариант	$\neg\mathrm{P}(\mathbf{f}(\mathrm{x}_2)) \vee \mathrm{R}(\mathrm{x}_2, \mathbf{g}(\mathrm{x}_2))$
резольвента	$\mathrm{R}(\mathrm{x}_3, \mathbf{g}(\mathrm{x}_3))$
вариант	$\neg\mathrm{R}(\mathrm{x}_4, \mathrm{u}_4)$
резольвента	$\square$

Следовательно, $\square$ резолютивно выводи́м из системы S

Резолютивный вывод

Другой пример

$$S = \left\{ \begin{array}{c} \neg \mathrm{P}(\mathbf{f}(\mathrm{x}), \mathrm{z}) \vee \neg \mathrm{P}(\mathrm{y}, \mathrm{y}) \\ \mathrm{P}(\mathrm{x}, \mathbf{f}(\mathrm{y})) \vee \mathrm{R}(\mathrm{y}) \\ \neg \mathrm{R}(\mathrm{y}) \end{array} \right\}$$

Резолютивный вывод $\square$ из S:

вариант	$\mathrm{P}(\mathrm{x}_1, \mathbf{f}(\mathrm{y}_1)) \vee \mathrm{R}(\mathrm{y}_1)$
вариант	$\neg \mathrm{R}(\mathrm{y}_2)$
резольвента	$\mathrm{P}(\mathrm{x}_3, \mathbf{f}(\mathrm{y}_3))$
вариант	$\neg \mathrm{P}(\mathbf{f}(\mathrm{x}_4), \mathrm{z}_4) \vee \neg \mathrm{P}(\mathrm{y}_4, \mathrm{y}_4)$
склейка	$\neg \mathrm{P}(\mathbf{f}(\mathrm{x}_5), \mathbf{f}(\mathrm{x}_5))$
резольвента	$\square$

Резолютивный вывод

Возможность использования всевозможных вариантов дизъюнктов наряду с самими дизъюнктами в резолютивном выводе настолько же важна, насколько и возможность использования резольвент и склеек

Например: $S = \{\neg \mathrm{P}(\mathrm{x}),\ \mathrm{P}(\mathbf{f}(\mathrm{x}))\}$

$\mathrm{НОУ}(\mathrm{P}(\mathrm{x}), \mathrm{P}(\mathbf{f}(\mathrm{x}))) = \emptyset$

Значит, у этих дизъюнктов нет ни одной резольвенты

При этом система S невыполнима:

у формул $\forall \mathrm{x}\ \neg \mathrm{P}(\mathrm{x})$ и $\forall \mathrm{x}\ \mathrm{P}(\mathbf{f}(\mathrm{x}))$ нет общих моделей

К вариантам дизъюнктов из S применимо правило резолюции:

$$\{\mathrm{x}_1/\mathbf{f}(\mathrm{x}_2)\} \in \mathrm{НОУ}(\mathrm{P}(\mathrm{x}_1), \mathrm{P}(\mathbf{f}(\mathrm{x}_2)))$$

Корректность использования всевозможных вариантов дизъюнктов обеспечивается следующей равносильностью:

$$\forall \mathrm{x}\ \varphi \sim \forall \mathrm{y}\ (\varphi\{\mathrm{x}/\mathrm{y}\}), \text{ если } <\ldots>$$

Резолютивный вывод

Резолютивный вывод успешен,
если он оканчивается пустым дизъюнктом ($\square$)

Успешный резолютивный вывод также называется
резолютивным опровержением:

- предположим, что
  исходная система дизъюнктов выполнима
- тогда система, к которой добавлены все дизъюнкты вывода,
  также выполнима (*это обосновывается дальше*)
- **противоречие:** среди добавленных дизъюнктов
  есть тождественно ложный ($\square$), а значит,
  расширенная система дизъюнктов невыполнима
- полученное противоречие
  опровергает выполнимость исходной системы
  (*доказывает невыполнимость методом «от противного»*)

Корректность резолютивного вывода

Теорема (о корректности резолютивного вывода)
Если из системы дизъюнктов S резолютивно выводи́м $\square$, то система S невыполнима

Доказательство.

Вариант D' любого дизъюнкта D равносилен D, а значит, $\mathrm{D} \models \mathrm{D}'$

По корректности правила резолюции:
если D'' — *резольвента* дизъюнктов D_1, D_2, то $\mathrm{D}_1, \mathrm{D}_2 \models \mathrm{D}''$

По корректности правила склейки:
если D''' — *склейка* дизъюнкта D_3, то $\mathrm{D}_3 \models \mathrm{D}'''$

Значит, по транзитивности логического следования,
любой дизъюнкт, выводимый из S,
является логическим следствием S, и в частности, $S \models \square$

При этом дизъюнкт $\square$ не имеет ни одной модели,
а значит, и система S не имеет ни одной модели ▼

Математическая логика и логическое программирование

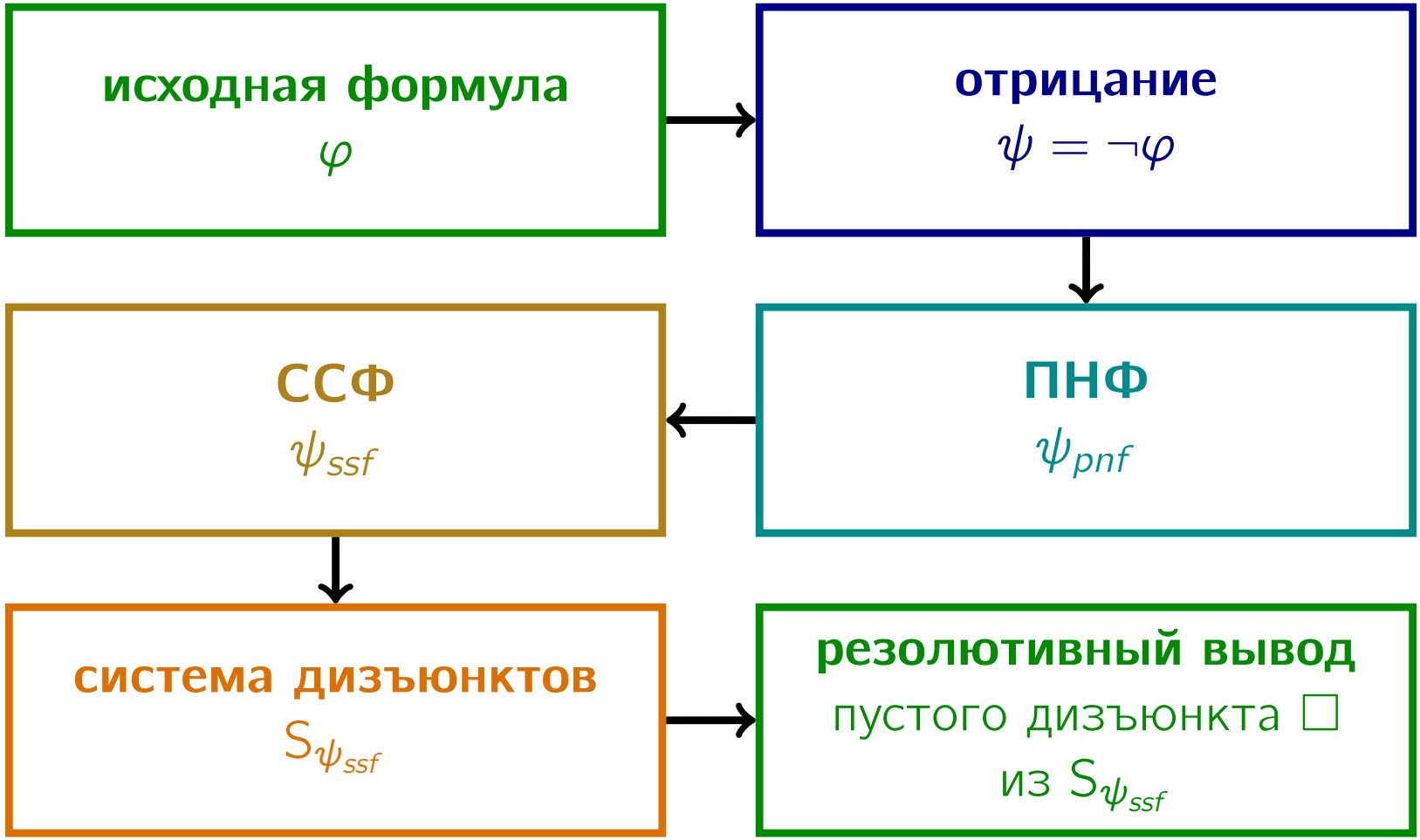
mk.cs.msu.ru → Лекционные курсы
→ Математическая логика и логическое программирование (3-й поток)

Блок 23

Обоснование общезначимости формулы методом резолюций (пример)

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

ВМК МГУ, 2023/2024, осенний семестр



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf} \quad \Leftrightarrow \quad \not\models S_{\psi_{ssf}}$$

$$\Leftarrow \quad \text{существует вывод } \square \text{ из } S_{\psi_{ssf}}$$

Примеры, которые использовались при обсуждении этапов метода резолюций, выбирались так, чтобы при их совмещении получился сквозной пример: обоснование общезначимости формулы

$$\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$

методом резолюций

Запишем получившееся обоснование от начала и до конца

Этап 1: поставить отрицание

$$\models \exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$

$$\Leftrightarrow$$

$$\not\models \neg\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$

$$\models \exists x\,(P(x)\,\&(\forall x\,P(x)\to\exists y\,R(x,y))\to\exists y\,R(x,y))$$

$$\Leftrightarrow$$

$$\not\models \neg\exists x\,(P(x)\,\&(\forall x\,P(x)\to\exists y\,R(x,y))\to\exists y\,R(x,y))$$

Этап 2: построить равносильную ПНФ

$$\neg\exists x\,(P(x)\,\&(\forall x\,P(x)\to\exists y\,R(x,y))\to\exists y\,R(x,y))$$

$$\sim (\textit{переименование переменных})$$

$$\neg\exists x\,(P(x)\,\&(\forall z\,P(z)\to\exists y\,R(x,y))\to\exists u\,R(x,u))$$

$$\sim (\textit{удаление импликаций})$$

$$\neg\exists x\,(\neg(P(x)\,\&(\neg\forall z\,P(z)\vee\exists y\,R(x,y)))\vee\exists u\,R(x,u))$$

$$\sim (\textit{продвижение отрицаний})$$

$$\forall x\,(P(x)\,\&(\exists z\,\neg P(z)\vee\exists y\,R(x,y))\,\&\,\forall u\,\neg R(x,u))$$

$$\sim (\textit{вынесение кванторов})$$

$$\forall x\,\exists z\,\exists y\,\forall u\,(P(x)\,\&(\neg P(z)\vee R(x,y))\,\&\,\neg R(x,u))$$

$$\sim (\textit{получение КНФ})$$

$$\forall x\,\exists z\,\exists y\,\forall u\,(P(x)\,\&(\neg P(z)\vee R(x,y))\,\&\,\neg R(x,u))$$

$$\models \exists x\,(P(x)\,\&(\forall x\,P(x)\to\exists y\,R(x,y))\to\exists y\,R(x,y))$$

$$\Leftrightarrow$$

$$\not\models \forall x\,\exists z\,\exists y\,\forall u\,(P(x)\,\&(\neg P(z)\vee R(x,y))\,\&\,\neg R(x,u))$$

Этап 3: построить ССФ, применив алгоритм сколемизации

$$\not\models \forall x\,\underline{\exists z}\,\underline{\exists y}\,\forall u\,(P(x)\,\&(\neg P(\underline{z})\vee R(x,\underline{y}))\,\&\,\neg R(x,u))$$

$$\Leftrightarrow$$

$$\not\models \forall x\,\forall u\,(P(x)\,\&(\neg P(\mathbf{f}(x))\vee R(x,\mathbf{g}(x)))\,\&\,\neg R(x,u))$$

Этап 4: перейти к системе дизъюнктов

$$\not\models \forall x\,\forall u\,(P(x)\,\&(\neg P(\mathbf{f}(x))\vee R(x,\mathbf{g}(x)))\,\&\,\neg R(x,u))$$

$$\Leftrightarrow$$

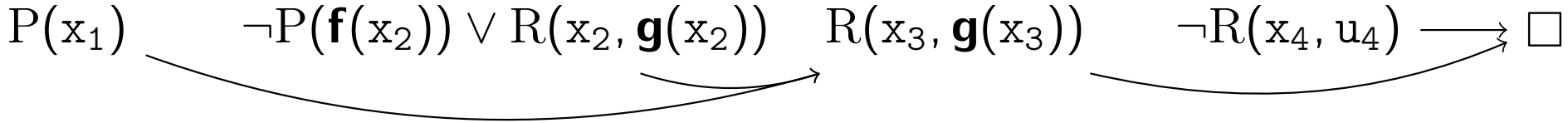
$$\not\models \{P(x),\ \neg P(\mathbf{f}(x))\vee R(x,\mathbf{g}(x)),\ \neg R(x,u)\}$$

$$\models \exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$

$$\Leftrightarrow$$

$$\not\models \{\mathrm{P}(\mathrm{x}),\quad \neg\mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x},\mathbf{g}(\mathrm{x})),\quad \neg\mathrm{R}(\mathrm{x},\mathrm{u})\}$$

Этап 5: резолютивно вывести $\square$



Оказалось, что $\square$ резолютивно выводи́м
из построенной системы дизъюнктов

Следовательно (по спектру доказанных ранее теорем),
исходная формула

$$\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$

общезначима

Математическая логика и логическое программирование

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

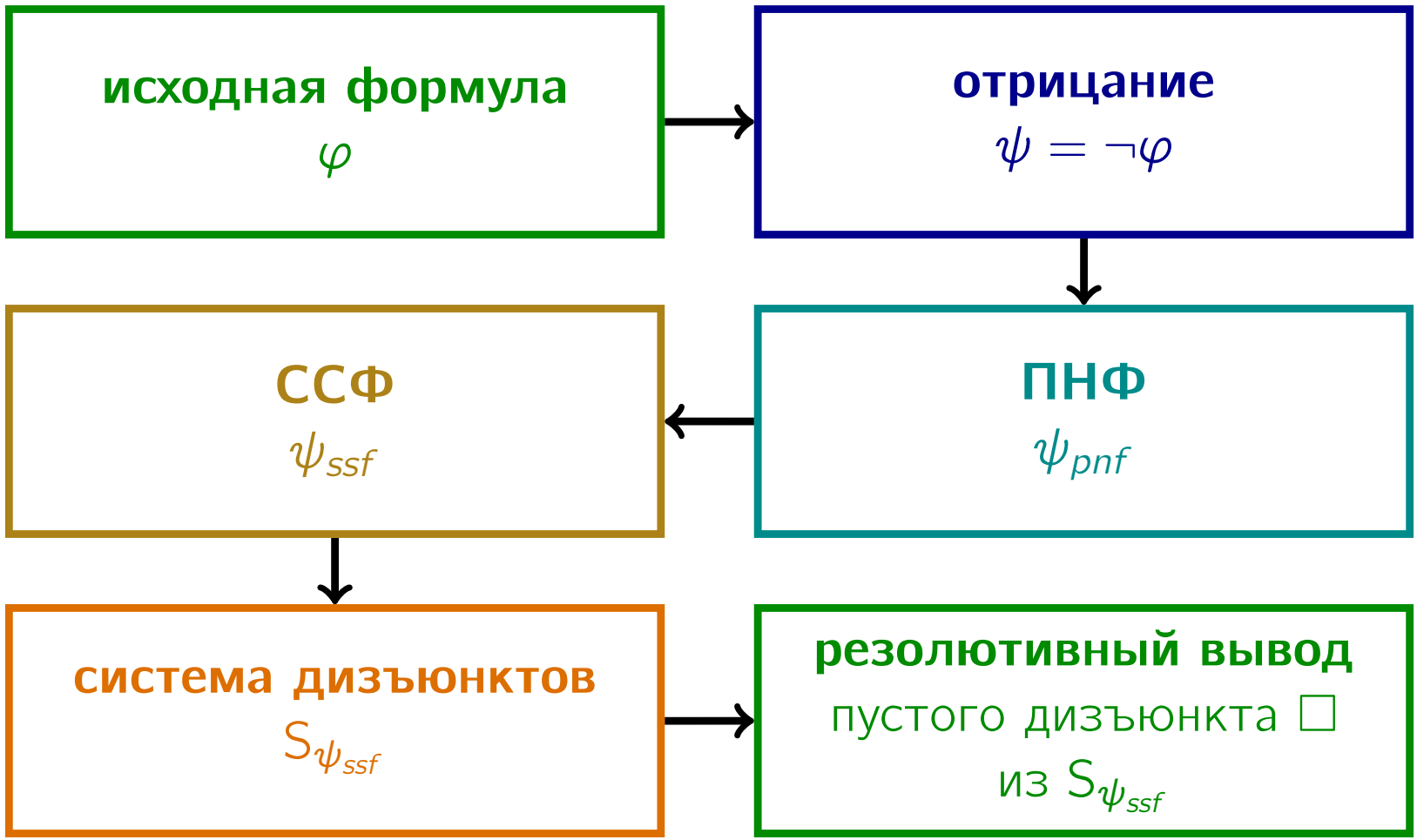
Блок 24

Эрбрановские интепретации
Теорема об эрбрановских интерпретациях

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

ВМК МГУ, 2023/2024, осенний семестр

Краткое вступление



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf} \quad \Leftrightarrow \quad \not\models S_{\psi_{ssf}}$$

$$\Leftarrow \quad \text{существует вывод } \square \text{ из } S_{\psi_{ssf}}$$

Последняя недостающая деталь метода — «$\Leftrightarrow$» на месте «$\Leftarrow$»

Краткое вступление

Временно забудем про метод резолюций
и попробуем выделить *общую часть* рассуждений «в лоб»
о невыполнимости системы дизъюнктов S в заданной интерпретации $\mathcal{I}$

Для примера рассмотрим такую систему: $S = \{\mathrm{P}(\mathbf{x}),\ \neg\mathrm{P}(\mathbf{f}(\mathbf{c}))\}$

Для обоснования невыполнимости S достаточно заметить,
что в любой модели $\mathcal{I}$ для S предмет, описываемый термом $\mathbf{f}(\mathbf{c})$,
и обладает свойством P ($\mathcal{I} \models \forall\mathbf{x}\,\mathrm{P}(\mathbf{x})$), и не обладает ($\mathcal{I} \models \neg\mathrm{P}(\mathbf{f}(\mathbf{c}))$)

В этом замечании не используются
природа предметной области $\mathcal{I}$ и устройство оценок $\bar{\mathbf{c}}$ и $\bar{\mathbf{f}}$,
и важно лишь то, каким термом задаётся «противоречивый» предмет

Если в интерпретации $\mathcal{I}$ заменить каждый предмет
множеством основных термов, описывающих этот предмет,
и сохранить устройство оценки P, то в результате получится

эрбрановская интерпретация

Эрбрановские интерпретации

Эрбрановской интерпретацией (или, по-другому, — $\mathcal{H}$-интерпретацией) сигнатуры $\sigma = \langle \text{Const}, \text{Func}, \text{Pred} \rangle$ называется интерпретация $\langle \mathcal{H}_\sigma, \overline{\text{Const}}_{\mathcal{H}}, \overline{\text{Func}}_{\mathcal{H}}, \overline{\text{Pred}} \rangle$, где:

- $\mathcal{H}_\sigma$ — эрбрановский универсум ($\mathcal{H}$-универсум): множество это множество всех основных термов сигнатуры
  - σ, если $\text{Const} \neq \emptyset$
  - $\langle \{\mathbf{c}_{\mathcal{H}}\}, \text{Func}, \text{Pred} \rangle$, если $\text{Const} = \emptyset$ ($\mathbf{c}_{\mathcal{H}}$ — эрбрановская константа)
- $\overline{\text{Const}}_{\mathcal{H}}(\mathbf{c}) = \mathbf{c}$ для любой константы $\mathbf{c}$ из Const
- $\overline{\text{Func}}_{\mathcal{H}}(\mathbf{f})(t_1, \ldots, t_n) = \mathbf{f}(t_1, \ldots, t_n)$ для любых $\mathbf{f}^{(n)} \in \text{Func}$ и $t_1, \ldots, t_n \in \mathcal{H}_\sigma$
- $\overline{\text{Pred}}$ — произвольная оценка предикатных символов

Первые три пункта означают, что $\mathcal{H}$-интерпретация — это интерпретация, построенная над свободной алгеброй термов

Все $\mathcal{H}$-интерпретации заданной сигнатуры отличаются друг от друга только выбором оценки $\overline{\text{Pred}}$

Эрбрановские интерпретации

Эрбрановский базис ($B_{\mathcal{H}}$) — это множество всех атомов, построенных над термами $\mathcal{H}$-универсума

$\mathcal{H}$-интерпретация $\mathcal{I}$ всегда и однозначно определяется тем, какие атомы из $B_{\mathcal{H}}$ в ней истинны и какие нет, то есть множеством

$$B^{\mathcal{I}} = \{A \mid A \in B_{\mathcal{H}},\ \mathcal{I} \models A\}$$

Например,

- $B^{\mathcal{I}} = \emptyset$: все основные атомы **ложны** в $\mathcal{I}$
- $B^{\mathcal{I}} = B_{\mathcal{H}}$: все основные атомы **истинны** в $\mathcal{I}$
- $B^{\mathcal{I}} = B^{\mathcal{J}'} \cap B^{\mathcal{J}''}$: в $\mathcal{I}$ истинны те и только те основные атомы, которые истинны в обеих интерпретациях $\mathcal{J}'$, $\mathcal{J}''$

Для удобного использования теоретико-множественной нотации иногда будем отождествлять $\mathcal{H}$-интерпретацию $\mathcal{I}$ с множеством $B^{\mathcal{I}}$

Теорема об эрбрановских интерпретациях

Система дизъюнктов выполнима тогда и только тогда, когда она имеет эрбрановскую модель

Доказательство.

Рассмотрим произвольную систему дизъюнктов S и докажем теорему для этой системы

($\Leftarrow$) Очевидно:
если S выполняется в некоторой эрбрановской интерпретации, то S выполняется хотя бы в одной интерпретации

($\Rightarrow$)

Пусть система S выполнима

Тогда для неё существует модель $\mathcal{I} = \langle \mathrm{D}, \overline{\mathrm{Const}}, \overline{\mathrm{Func}}, \overline{\mathrm{Pred}} \rangle$

Построим по интерпретации $\mathcal{I}$
эрбрановскую модель $\mathcal{I}_{\mathcal{H}}$ для S
той же сигнатуры σ

Теорема об эрбрановских интерпретациях

Доказательство. ($\Rightarrow$) $\mathcal{I} = \langle \mathsf{D}, \overline{\mathsf{Const}}, \overline{\mathsf{Func}}, \overline{\mathsf{Pred}} \rangle \mapsto \mathcal{I}_{\mathcal{H}}$

Если $\mathsf{Const} = \emptyset$, то добавим и произвольно оценим константу $\mathbf{c}_{\mathcal{H}}$ в $\mathcal{I}$

Поставим такое соответствие $\alpha : \mathcal{H}_\sigma \to D$:

$\alpha(t)$ — значение терма t в $\mathcal{I}$

Зададим оценку $\overline{\overline{\mathrm{P}}}$ каждого предикатного символа P в $\mathcal{I}_{\mathcal{H}}$ так:

$$\overline{\overline{\mathrm{P}}}(t_1, \ldots, t_k) = \overline{\mathrm{P}}(\alpha(t_1), \ldots, \alpha(t_k))$$

То же самое другими словами — зададим интерпретацию $\mathcal{I}_{\mathcal{H}}$ так:

$$\mathcal{I}_{\mathcal{H}} = \{\mathrm{P}(t_1, \ldots, t_k) \mid \mathrm{P}(t_1, \ldots, t_k) \in B_{\mathcal{H}},\ \overline{\mathrm{P}}(\alpha(t_1), \ldots, \alpha(t_k)) = \mathbb{t}\}$$

Покажем, что такая интерпретация $\mathcal{I}_{\mathcal{H}}$ является моделью для S

Но для начала приведём пример, чтобы стало *понятнее*,
как соотносятся $\mathcal{I}$ и $\mathcal{I}_{\mathcal{H}}$

Теорема об эрбрановских интерпретациях

Доказательство. ($\Rightarrow$) $\mathcal{I} = \langle \mathsf{D}, \overline{\mathsf{Const}}, \overline{\mathsf{Func}}, \overline{\mathsf{Pred}} \rangle \mapsto \mathcal{I}_{\mathcal{H}}$

Пример: рассмотрим интерпретацию $\mathcal{I}$ сигнатуры $\langle \{0\}, \{\mathbf{s}^{(1)}\}, \{<^{(2)}\} \rangle$ с *естественным* арифметическим устройством:

- предметная область — множество всех целых чисел
- $\overline{0}$ — число 0; $\quad \overline{\mathbf{s}}(n) = n + 1$;
  $\overline{<}$ — отношение строгого неравенства чисел

Тогда:

- $\mathcal{H}_\sigma = \{0, \quad \mathbf{s}(0), \quad \mathbf{s}(\mathbf{s}(0)), \quad \mathbf{s}(\mathbf{s}(\mathbf{s}(0))), \quad \dots\}$
- $\mathcal{I}_{\mathcal{H}} =$

$$\begin{Bmatrix} 0 < \mathbf{s}(0), & \mathbf{s}(0) < \mathbf{s}(\mathbf{s}(0)), & \mathbf{s}(\mathbf{s}(0)) < \mathbf{s}(\mathbf{s}(\mathbf{s}(0))), & \dots \\ 0 < \mathbf{s}(\mathbf{s}(0)), & \mathbf{s}(0) < \mathbf{s}(\mathbf{s}(\mathbf{s}(0))), & \mathbf{s}(\mathbf{s}(0)) < \mathbf{s}(\mathbf{s}(\mathbf{s}(\mathbf{s}(0)))), & \dots \\ 0 < \mathbf{s}(\mathbf{s}(\mathbf{s}(0))), & \mathbf{s}(0) < \mathbf{s}(\mathbf{s}(\mathbf{s}(\mathbf{s}(0)))), & \mathbf{s}(\mathbf{s}(0)) < \mathbf{s}(\mathbf{s}(\mathbf{s}(\mathbf{s}(\mathbf{s}(0))))), & \dots \\ \dots & \dots & \dots & \dots \end{Bmatrix}$$

Теорема об эрбрановских интерпретациях

Доказательство. ($\Rightarrow$) $\mathcal{I} = \langle \mathrm{D}, \overline{\mathrm{Const}}, \overline{\mathrm{Func}}, \overline{\mathrm{Pred}} \rangle \mapsto \mathcal{I}_{\mathcal{H}}$

Предположим от противного, что $\mathcal{I}_{\mathcal{H}} \not\models S$

Тогда в S содержится дизъюнкт D, такой что $\mathcal{I}_{\mathcal{H}} \not\models D$

Пусть, для ясности, $D = \forall \widetilde{\mathrm{x}^{\mathrm{n}}} \, (A_1 \vee \cdots \vee A_k \vee \neg B_1 \vee \cdots \vee \neg B_m)$,
где $A_1, \ldots, A_k, B_1, \ldots, B_m$ — атомы

Тогда существуют термы $t_1, \ldots, t_n \in \mathcal{H}_\sigma$, такие что

$$\mathcal{I}_{\mathcal{H}} \not\models A_1[t_1, \ldots, t_n] \quad \ldots \quad \mathcal{I}_{\mathcal{H}} \not\models A_k[t_1, \ldots, t_n]$$
$$\mathcal{I}_{\mathcal{H}} \models B_1[t_1, \ldots, t_n] \quad \ldots \quad \mathcal{I}_{\mathcal{H}} \models B_m[t_1, \ldots, t_n]$$

По заданию интерпретации $\mathcal{I}_{\mathcal{H}}$, верно и

$$\mathcal{I} \not\models A_1[\alpha(t_1), \ldots, \alpha(t_n)] \quad \ldots \quad \mathcal{I} \not\models A_k[\alpha(t_1), \ldots, \alpha(t_n)]$$
$$\mathcal{I} \models B_1[\alpha(t_1), \ldots, \alpha(t_n)] \quad \ldots \quad \mathcal{I} \models B_m[\alpha(t_1), \ldots, \alpha(t_n)]$$

Следовательно, $\mathcal{I} \not\models \forall \widetilde{\mathrm{x}^{\mathrm{n}}} \, (A_1 \vee \cdots \vee A_k \vee \neg B_1 \vee \cdots \vee \neg B_m)$,
что *противоречит* выбору $\mathcal{I}$ как **модели** для S

Значит, предположение «$\mathcal{I}_{\mathcal{H}} \not\models S$» неверно, то есть $\mathcal{I}_{\mathcal{H}} \models S$ ▼

Математическая логика и логическое программирование

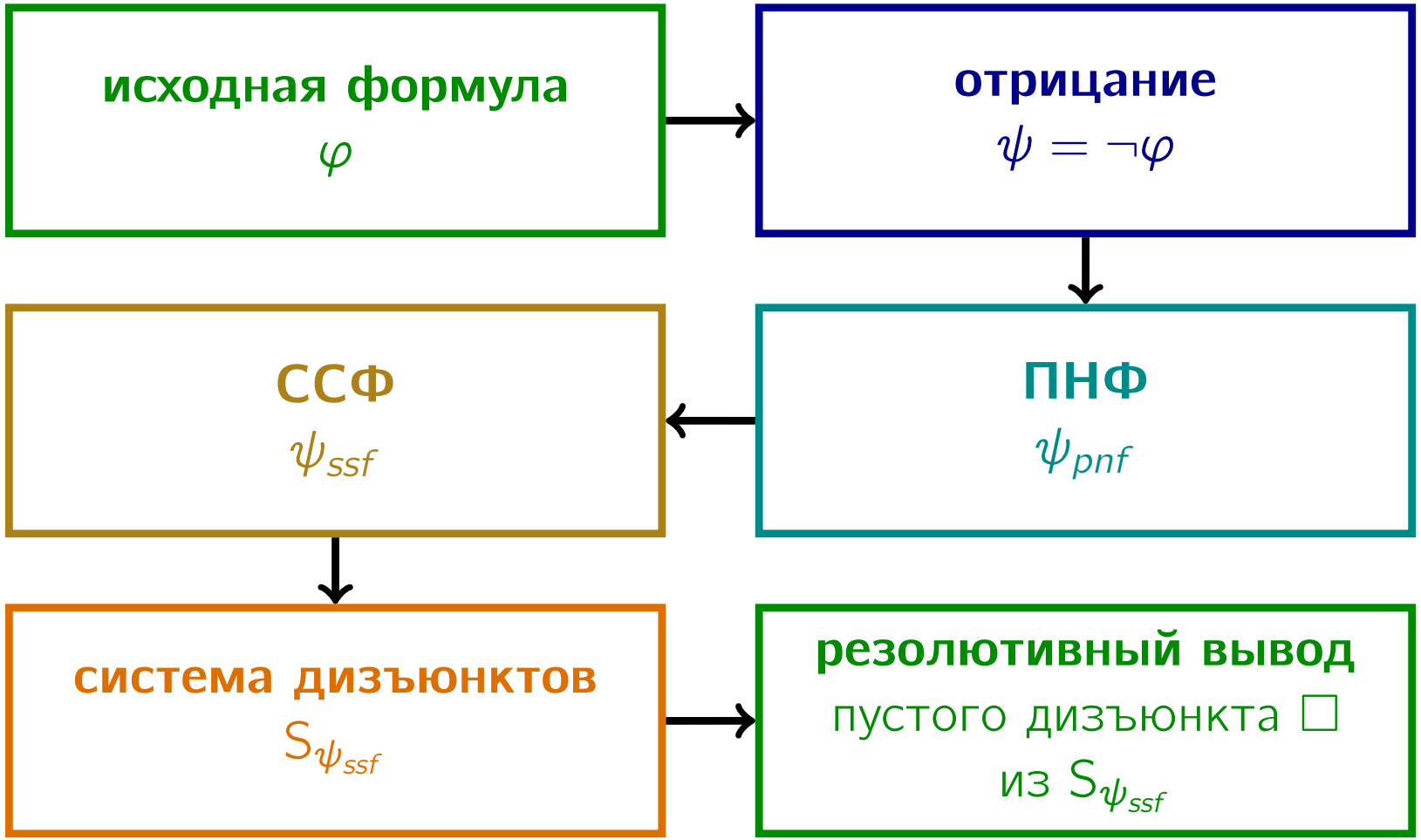
mk.cs.msu.ru → Лекционные курсы
→ Математическая логика и логическое программирование (3-й поток)

Блок 25

Теорема Эрбрана
Полнота резолютивного вывода

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

ВМК МГУ, 2023/2024, осенний семестр



$$\models \varphi \quad \Leftrightarrow \quad \not\models \psi \quad \Leftrightarrow \quad \not\models \psi_{pnf} \quad \Leftrightarrow \quad \not\models \psi_{ssf} \quad \Leftrightarrow \quad \not\models S_{\psi_{ssf}}$$

$$\Leftarrow \quad \text{существует вывод } \square \text{ из } S_{\psi_{ssf}}$$

Покажем, что резолютивный вывод **полон**:
справедливо не только «$\Leftarrow$», но и «$\Rightarrow$»

Общая схема обоснования полноты

Рассмотрим невыполнимую систему дизъюнктов S

Резолютивная выводимость $\square$ из S будет обоснована так:

1. Существует конечное невыполнимое множество $\mathcal{G}$ основных примеров дизъюнктов из S (*теорема Эрбрана*)

2. Существует резолютивный вывод $\square$ из $\mathcal{G}$ (*лемма об основных дизъюнктах*)

3. Каждый шаг вывода $\square$ из $\mathcal{G}$ можно преобразовать в «аналогичный» шаг вывода из S (две *леммы о подъёме*)

4. Следовательно, весь вывод $\square$ из $\mathcal{G}$ можно преобразовать в вывод $\square$ из S (*теорема о полноте резолютивного вывода*)

Теорема Эрбрана

Система дизъюнктов невыполнима

$\Leftrightarrow$

существует конечное невыполнимое множество основных примеров дизъюнктов этой системы

Теорема Эрбрана (доказательство)

Система дизъюнктов S (*сигнатуры* σ) невыполнима

$\Leftrightarrow$ (теорема об эрбрановских интерпретациях)

S не выполняется ни в одной $\mathcal{H}$-интерпретации

$\Leftrightarrow$ (определение выполнимости и семантика квантора $\forall$)

Для каждой $\mathcal{H}$-интерпретации $\mathcal{I}$ существуют дизъюнкт $\forall\widetilde{\mathrm{x}^{\mathrm{n}}}\ \mathrm{D} \in S$ и предметы $t_1, \ldots, t_n \in \mathcal{H}_\sigma$, такие что $\mathcal{I} \not\models \mathrm{D}[t_1, \ldots, t_n]$

$\Leftrightarrow$ (устройство $\mathcal{H}$-интерпретаций)

Для каждой $\mathcal{H}$-интерпретации $\mathcal{I}$ существует основной пример D' дизъюнкта $\mathrm{D} \in S$, такой что $\mathcal{I} \not\models \mathrm{D}'$

$\Leftrightarrow$ (то же другими словами)

Множество $\mathcal{G}_S$ всех основных примеров дизъюнктов из S не выполняется ни в одной $\mathcal{H}$-интерпретации

$\Leftrightarrow$ (теорема об эрбрановских интерпретациях)

Система $\mathcal{G}_S$ невыполнима

Теорема Эрбрана (доказательство)

Вспомним следствие из теоремы компактности Мальцева:

$\not\models \Gamma \quad \Leftrightarrow$
существует конечное невыполнимое подмножество Γ' множества Γ

Приме́ним это следствие к множеству $\mathcal{G}_S$:

$\not\models S \Leftrightarrow \ldots \Leftrightarrow \not\models \mathcal{G}_S$
$\Leftrightarrow$
существует **конечное** невыполнимое подмножество $\mathcal{G}$ множества $\mathcal{G}_S$ ▼

Лемма об основных дизъюнктах

Из любой конечной невыполнимой системы основных дизъюнктов резолютивно выводи́м пустой дизъюнкт

Доказательство.

Рассмотрим произвольную конечную невыполнимую систему основных дизъюнктов S

Докажем лемму индукцией по числу $\|S\|$ различных атомов, содержащихся в дизъюнктах из S

База: $\|S\| = 0$

$$\left.\begin{array}{rr} \not\models S & \Rightarrow \quad S \neq \emptyset \\ & \|S\| = 0 \end{array}\right\} \Rightarrow\ S = \{\square\} \ \Rightarrow\ \square \text{ резолютивно выводи́м из } S$$

Лемма об основных дизъюнктах

Доказательство.

Переход: $\not\models S$, $\|S\| = N > 0$; $\quad S \overset{?}{\rightsquigarrow} \square$

Индуктивное предположение: $\not\models \widetilde{S}$ и $\|\widetilde{S}\| < N \quad \Rightarrow \quad \widetilde{S} \rightsquigarrow \square$

Перейдём от S к системе дизъюнктов S',
удалив все дизъюнкты вида $D \vee A \vee \neg A$

$\not\models S$, и все удаляемые дизъюнкты общезначимы,
а значит, $\not\models S'$, и достаточно предложить вывод $\square$ из S'

Перейдём от S' к системе S'' так: пока это возможно,
будем заменять дизъюнкты вида $D \vee L \vee L$ на их склейки $D \vee L$

$\not\models S'$, и дизъюнкты заменяются на равносильные выводимые
а значит, $\not\models S''$, и достаточно предложить вывод $\square$ из S''

Если $\|S''\| < N$, то обоснование завершено

Предположим теперь, что $\|S''\| = N$

Лемма об основных дизъюнктах

Доказательство.

Переход: $\not\models S''$, $\|S''\| = N > 0$,

и в S'' нет дизъюнктов вида $D \vee L \vee \neg L$ и $D \vee L \vee L$; $\quad S'' \overset{?}{\rightsquigarrow} \square$

Индуктивное предположение: $\not\models \widetilde{S}$, $\|\widetilde{S}\| < N \quad \Rightarrow \quad \widetilde{S} \rightsquigarrow \square$

Произвольно выберем атом A,
содержащийся хотя бы в одном дизъюнкте из S''

Разобьём S'' на три множества:

- S_+ — все дизъюнкты из S'', содержащие A без отрицания
- S_- — все дизъюнкты из S'', содержащие $\neg A$
- $S_\times$ — все остальные дизюнкты из S''

Построим все резольвенты по контрарной паре A, $\neg A$:

$$S_r = \{D_1 \vee D_2 \mid D_1 \vee A \in S_+,\ D_2 \vee \neg A \in S_-\}$$

Достаточно показать, что $\square$ выводи́м из системы $S''' = S_r \cup S_\times$

$\|S'''\| = \|S''\| - 1 = N - 1 < N$, а значит, по предположению индукции, достаточно обосновать соотношение $\not\models S'''$

Лемма об основных дизъюнктах

Доказательство.

Переход:

$\not\models S''$: S_+ $\boxed{\ldots\ D_+ \vee A\ \ldots}$ S_- $\boxed{\ldots\ D_- \vee \neg A\ \ldots}$ $S_\times$ $\boxed{\text{дизъ. без } A}$

$\not\models ?\ S'''$: S_r $\boxed{\ldots\ D_+ \vee D_-\ \ldots}$ $S_\times$ $\boxed{\text{дизъ. без } A}$

Рассмотрим произвольную $\mathcal{H}$-интерпретацию $\mathcal{I}$

По теореме об эрбрановских интерпретациях,
достаточно обосновать соотношение $\mathcal{I} \not\models S'''$

Так как система S'' невыполнима, верно $\mathcal{I} \not\models S''$

Случай 1: $\mathcal{I} \not\models S_\times$. Так как $S_\times \subseteq S'''$, верно и $\mathcal{I} \not\models S'''$

Случай 2: $\mathcal{I} \models S_\times$, $A \in \mathcal{I}$

$A \in \mathcal{I} \Rightarrow \mathcal{I} \models S_+$

$\mathcal{I} \models S_+ \cup S_\times$ и $\mathcal{I} \not\models S_+ \cup S_\times \cup S_- \Rightarrow \mathcal{I} \not\models S_-$
$\Rightarrow$ в S_- содержится дизъюнкт $D_- \vee \neg A$, такой что $\mathcal{I} \not\models D_- \vee \neg A$
$\Rightarrow \mathcal{I} \not\models D_-$

Лемма об основных дизъюнктах

Доказательство.

Переход:

$\not\models S''$: S_+ $\boxed{\ldots\ D_+ \vee A\ \ldots}$ S_- $\boxed{\ldots\ D_- \vee \neg A\ \ldots}$ $S_\times$ $\boxed{\text{дизъ. без } A}$

$\mathcal{I} \not\models ?\ S'''$: S_r $\boxed{\ldots\ D_+ \vee D_-\ \ldots}$ $S_\times$ $\boxed{\text{дизъ. без } A}$

Случай 2: $\mathcal{I} \models S_\times$, $A \in \mathcal{I}$; получено: $\mathcal{I} \not\models D_-$, $D_- \vee \neg A \in S_-$

Рассмотрим $\mathcal{H}$-интерпретацию $\mathcal{J} = \mathcal{I} \setminus \{A\}$

$\mathcal{I} \models S_\times$ и в дизъюнктах из $S_\times$ не содержится $A \Rightarrow \mathcal{J} \models S_\times$

$\not\models S'' \Rightarrow \mathcal{J} \not\models S''$

$A \notin \mathcal{J}$ и дизъюнкты из S_- имеют вид $D \vee \neg A \Rightarrow \mathcal{J} \models S_-$

$\mathcal{J} \models S_- \cup S_\times$ и $\mathcal{J} \not\models S_- \cup S_\times \cup S_+ \Rightarrow \mathcal{J} \not\models S_+ \Rightarrow$
в S_+ содержится дизъюнкт $D_+ \vee A$, такой что $\mathcal{J} \not\models D_+ \vee A \Rightarrow \mathcal{J} \not\models D_+$

$\mathcal{J} \not\models D_+$ и в D_+ не содержится атом $A \Rightarrow \mathcal{I} \not\models D_+$

$\mathcal{I} \not\models D_+$ и $\mathcal{I} \not\models D_- \Rightarrow \mathcal{I} \not\models D_+ \vee D_-$

$\mathcal{I} \not\models D_+ \vee D_-$ и $D_+ \vee D_- \in S_r \Rightarrow \mathcal{I} \not\models S_r \Rightarrow \mathcal{I} \not\models S'''$

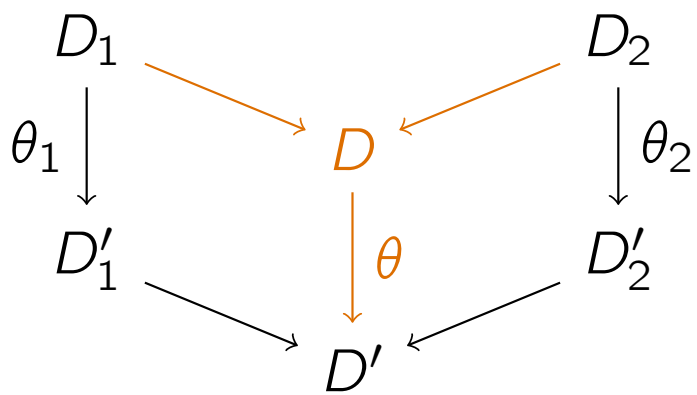
Случай 3: $\mathcal{I} \models S_\times$, $A \notin \mathcal{I}$ — аналогичен случаю 2 ▼

Лемма о подъёме для правила резолюции

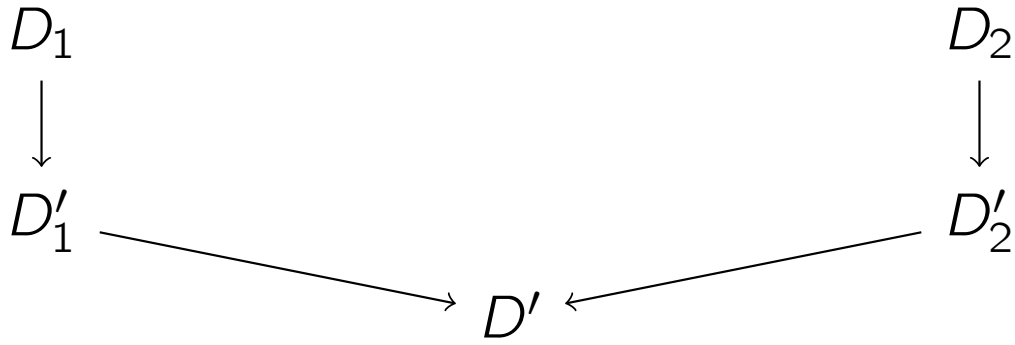
Пусть:

- D_1, D_2 — **дизъюнкты, и** $\mathrm{Var}_{D_1} \cap \mathrm{Var}_{D_2} = \emptyset$**;**
- D'_1, D'_2 — **основные примеры дизъюнктов** D_1, D_2 **соответственно;**
- D' — **резольвента дизъюнктов** D'_1, D'_2**.**

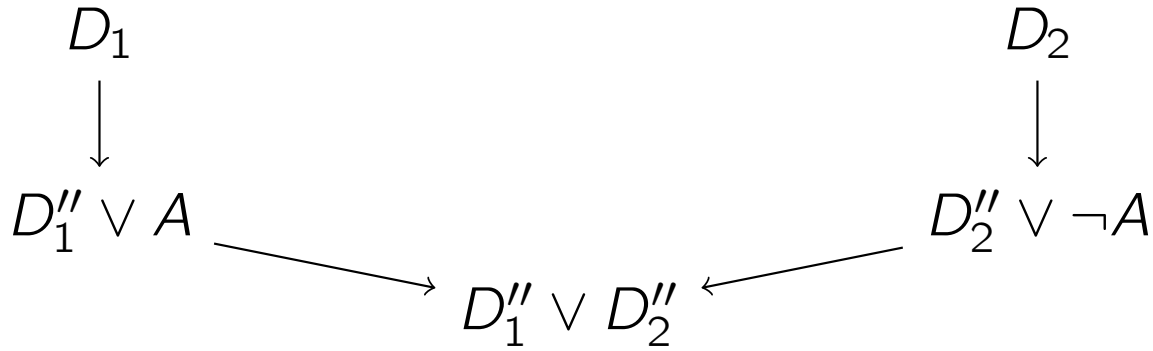
Тогда существует резольвента D **дизъюнктов** D_1, D_2, **примером которой является** D'**.**



Лемма о подъёме ... (доказательство)

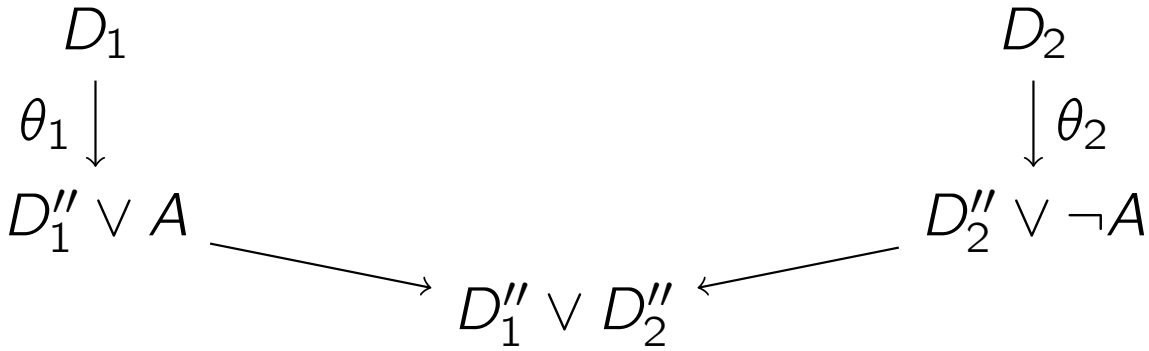


Лемма о подъёме ... (доказательство)



Выделим в D_1', D_2' контрарную пару A, $\neg A$ для резольвенты D'

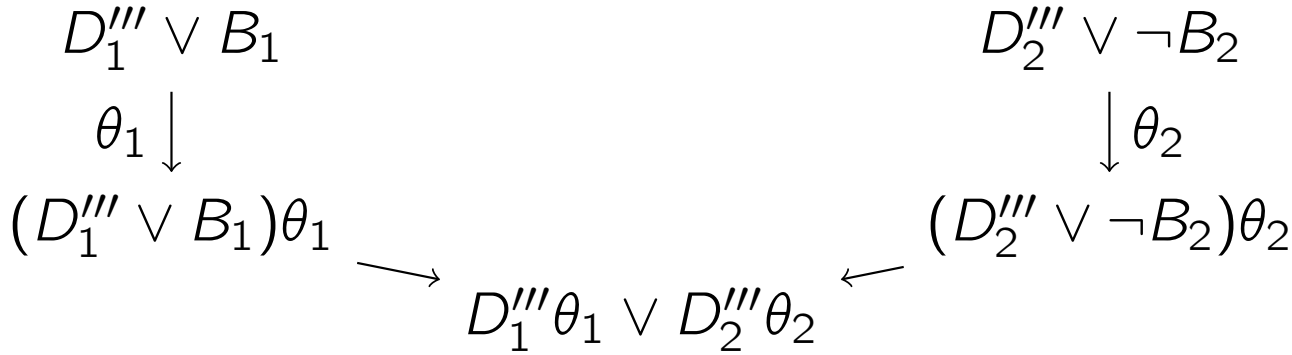
Лемма о подъёме ... (доказательство)



Выделим в D_1', D_2' контрарную пару A, $\neg A$ для резольвенты D'

Пусть, для ясности, $D_1' = D_1\theta_1$ и $D_2' = D_2\theta_2$

Лемма о подъёме ... (доказательство)

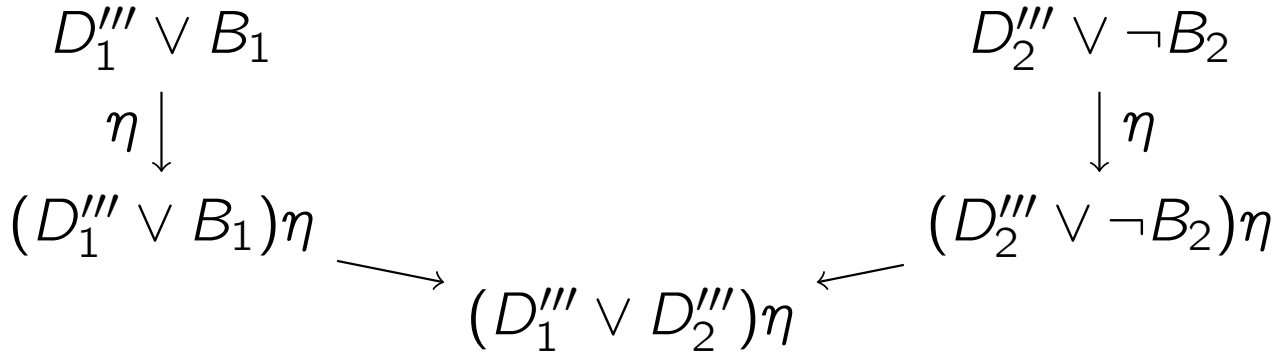


Выделим в D_1', D_2' контрарную пару A, $\neg A$ для резольвенты D'

Пусть, для ясности, $D_1' = D_1\theta_1$ и $D_2' = D_2\theta_2$

Выделим в D_1, D_2 литеры B_1, $\neg B_2$, породившие контрарную пару

Лемма о подъёме ... (доказательство)



Выделим в D_1', D_2' контрарную пару A, $\neg A$ для резольвенты D'

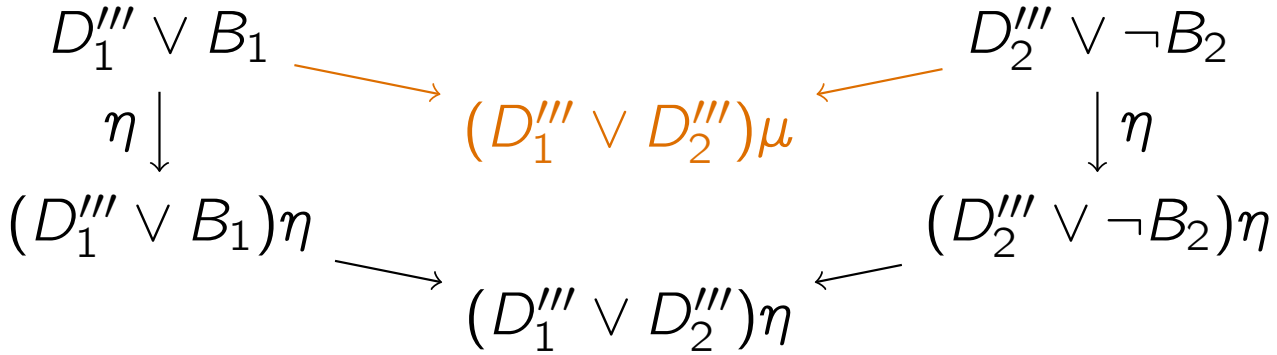
Пусть, для ясности, $D_1' = D_1\theta_1$ и $D_2' = D_2\theta_2$

Выделим в D_1, D_2 литеры B_1, $\neg B_2$, породившие контрарную пару

Без ограничения общности положим, что $\mathrm{Dom}_{\theta_1} \cap \mathrm{Dom}_{\theta_2} = \emptyset$

Тогда для подстановки $\eta = \theta_1 \cup \theta_2$ верно $D_1' = D_1\eta$ и $D_2' = D_2\eta$

Лемма о подъёме ... (доказательство)



Выделим в D_1', D_2' контрарную пару A, $\neg A$ для резольвенты D'

Пусть, для ясности, $D_1' = D_1\theta_1$ и $D_2' = D_2\theta_2$

Выделим в D_1, D_2 литеры B_1, $\neg B_2$, породившие контрарную пару

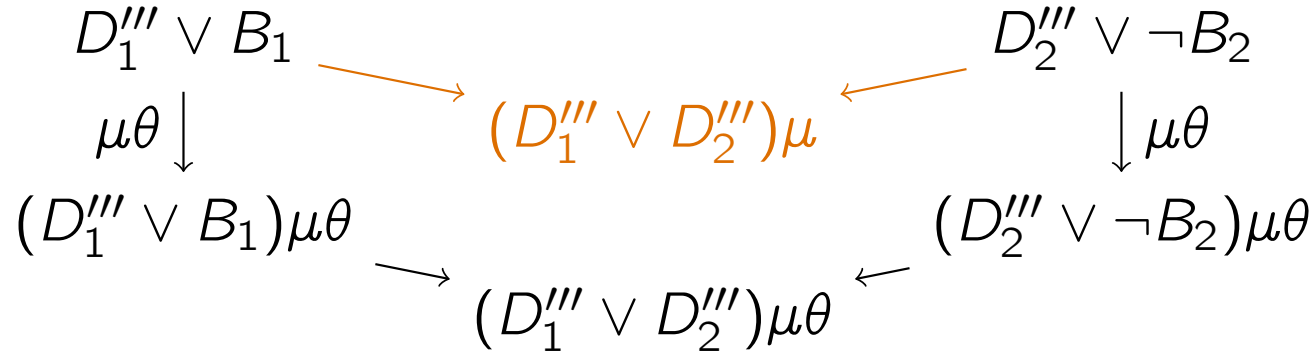
Без ограничения общности положим, что $\mathrm{Dom}_{\theta_1} \cap \mathrm{Dom}_{\theta_2} = \emptyset$

Тогда для подстановки $\eta = \theta_1 \cup \theta_2$ верно $D_1' = D_1\eta$ и $D_2' = D_2\eta$

По теореме об унификации,
существует наиболее общий унификатор μ атомов B_1, B_2

Значит, $(D_1''' \vee D_2''')\mu$ — резольвента дизъюнктов D_1, D_2

Лемма о подъёме ... (доказательство)



Выделим в D_1', D_2' контрарную пару A, $\neg A$ для резольвенты D'

Пусть, для ясности, $D_1' = D_1\theta_1$ и $D_2' = D_2\theta_2$

Выделим в D_1, D_2 литеры B_1, $\neg B_2$, породившие контрарную пару

Без ограничения общности положим, что $\mathrm{Dom}_{\theta_1} \cap \mathrm{Dom}_{\theta_2} = \emptyset$

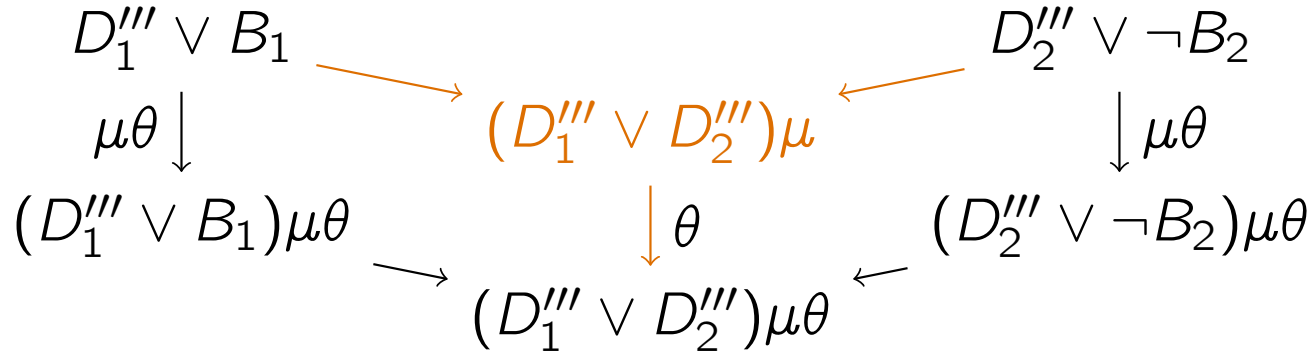
Тогда для подстановки $\eta = \theta_1 \cup \theta_2$ верно $D_1' = D_1\eta$ и $D_2' = D_2\eta$

По теореме об унификации,
существует наиболее общий унификатор μ атомов B_1, B_2

Значит, $(D_1''' \vee D_2''')\mu$ — резольвента дизъюнктов D_1, D_2

По определению наиболее общего унификатора,
существует подстановка θ, такая что $\eta = \mu\theta$

Лемма о подъёме ... (доказательство)



Выделим в D_1', D_2' контрарную пару A, $\neg A$ для резольвенты D'

Пусть, для ясности, $D_1' = D_1\theta_1$ и $D_2' = D_2\theta_2$

Выделим в D_1, D_2 литеры B_1, $\neg B_2$, породившие контрарную пару

Без ограничения общности положим, что $\mathrm{Dom}_{\theta_1} \cap \mathrm{Dom}_{\theta_2} = \emptyset$

Тогда для подстановки $\eta = \theta_1 \cup \theta_2$ верно $D_1' = D_1\eta$ и $D_2' = D_2\eta$

По теореме об унификации,
существует наиболее общий унификатор μ атомов B_1, B_2

Значит, $(D_1''' \vee D_2''')\mu$ — резольвента дизъюнктов D_1, D_2

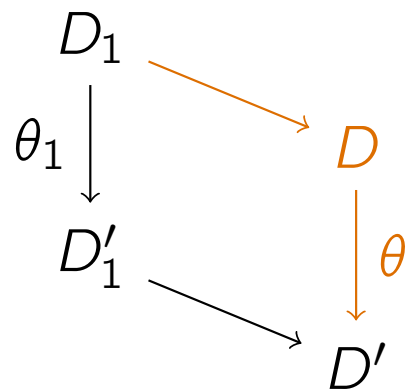
По определению наиболее общего унификатора,
существует подстановка θ, такая что $\eta = \mu\theta$, и $D' = D\theta$ ▼

Лемма о подъёме для правила склейки

Пусть:

- D_1, D'_1 — **дизъюнкт и его основной пример;**
- D' — **склейка дизъюнкта D'_1.**

Тогда существует склейка D дизъюнкта D_1, примером которой является D'.



Доказательство. Аналогично предыдущей лемме

Теорема о полноте резолютивного вывода

Из любой невыполнимой системы дизъюнктов резолютивно выводи́м пустой дизъюнкт

Доказательство.

Рассмотрим произвольную невыполнимую систему дизъюнктов S

По теореме Эрбрана, существует конечное невыполнимое множество $\mathcal{G}$, содержащее только основные примеры дизъюнктов из S

По лемме об основных дизъюнктах,
существует вывод $D'_1, \ldots, D'_n, \square$ из $\mathcal{G}$

Теорема о полноте резолютивного вывода

Из любой невыполнимой системы дизъюнктов резолютивно выводи́м пустой дизъюнкт

Доказательство. ($D'_1, \ldots, D'_n, \square$ — вывод из $\mathcal{G}$)

Рассмотрим такую последовательность дизъюнктов $\mathfrak{S} = (D_1, \ldots, D_n, \square)$:

- если D'_i — **пример** дизъюнкта D из S, то D_i — вариант D
- если D'_i — **резольвента** D'_j и D'_k ($j < i$, $k < i$), то D_i — резольвента D_j и D_k, примером которой является D'_j
- если D'_i — **склейка** D'_j ($j < i$), то D_i — склейка D_j, примером которой является D'_j
- подстановки (переименования и унификаторы) выберем так, чтобы множества Var_{D_i} попарно не пересекались

Корректность задания $\mathfrak{S}$ обеспечивается леммами о подъёме

По определению резолютивного вывода, $\mathfrak{S}$ — вывод $\square$ из S ▼

Математическая логика и логическое программирование

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

Блок 26

Иллюстрация полноты резолютивного вывода

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

ВМК МГУ, 2023/2024, осенний семестр

Ранее с использованием метода резолюций
была доказана общезначимость формулы

$$\exists \mathrm{x}\,(\mathrm{P}(\mathrm{x})\,\&(\forall \mathrm{x}\,\mathrm{P}(\mathrm{x}) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y})) \to \exists \mathrm{y}\,\mathrm{R}(\mathrm{x},\mathrm{y}))$$

Это обоснование основывалось на резолютивном выводе $\square$
из системы дизъюнктов

$$S = \{\mathrm{P}(\mathrm{x}),\quad \neg\mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x},\mathbf{g}(\mathrm{x})),\quad \neg\mathrm{R}(\mathrm{x},\mathrm{u})\},$$

соответствующей этой формуле

Проиллюстрируем схему обоснования полноты метода резолюций
на этой системе

Известно, что эта система S невыполнима

По теореме Эрбрана, существует конечное невыполнимое множество
основных примеров дизъюнктов этой системы

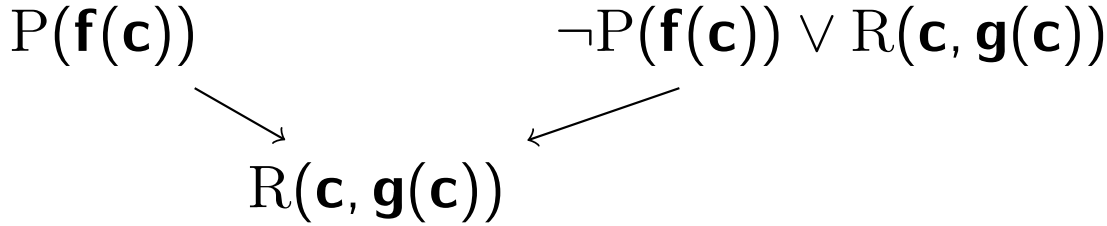
Например, невыполнимо такое множество примеров:

$$\mathcal{G} = \{\mathrm{P}(\mathbf{f}(\mathbf{c})),\quad \neg\mathrm{P}(\mathbf{f}(\mathbf{c})) \vee \mathrm{R}(\mathbf{c},\mathbf{g}(\mathbf{c})),\quad \neg\mathrm{R}(\mathbf{c},\mathbf{g}(\mathbf{c}))\}$$

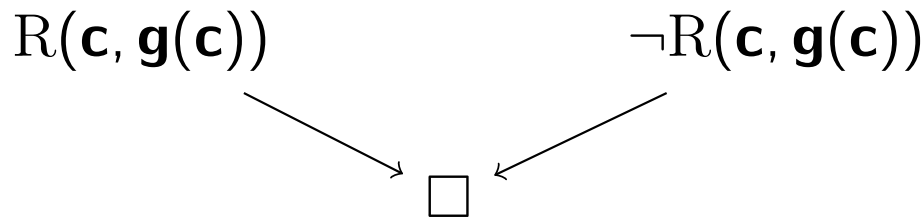
$$S = \{\mathrm{P}(\mathtt{x}), \quad \neg\mathrm{P}(\mathbf{f}(\mathtt{x})) \vee \mathrm{R}(\mathtt{x}, \mathbf{g}(\mathtt{x})), \quad \neg\mathrm{R}(\mathtt{x}, \mathtt{u})\},$$
$$\mathcal{G} = \{\mathrm{P}(\mathbf{f}(\mathbf{c})), \quad \neg\mathrm{P}(\mathbf{f}(\mathbf{c})) \vee \mathrm{R}(\mathbf{c}, \mathbf{g}(\mathbf{c})), \quad \neg\mathrm{R}(\mathbf{c}, \mathbf{g}(\mathbf{c}))\}$$

Действуя согласно лемме об основных дизъюнктах, построим вывод $\square$ из $\mathcal{G}$:

- Постро́им все резольвенты относительно $\mathrm{P}(\mathbf{f}(\mathbf{c}))$:



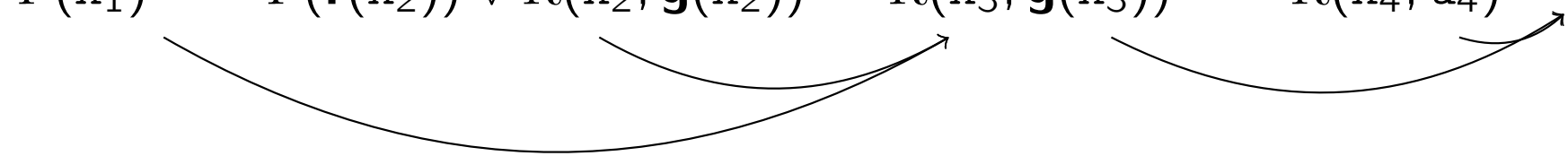
- Для построенных резольвент и дизъюнктов, не содержащих $\mathrm{P}(\mathbf{f}(\mathbf{c}))$, постро́им все резольвенты относительно $\mathrm{R}(\mathbf{c}, \mathbf{g}(\mathbf{c}))$:



$$S = \{\mathrm{P}(\mathrm{x}), \quad \neg\mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x}, \mathbf{g}(\mathrm{x})), \quad \neg\mathrm{R}(\mathrm{x}, \mathrm{u})\},$$
$$\mathcal{G} = \{\mathrm{P}(\mathbf{f}(\mathbf{c})), \quad \neg\mathrm{P}(\mathbf{f}(\mathbf{c})) \vee \mathrm{R}(\mathbf{c}, \mathbf{g}(\mathbf{c})), \quad \neg\mathrm{R}(\mathbf{c}, \mathbf{g}(\mathbf{c}))\}$$

Получен такой резолютивный вывод $\square$ из $\mathcal{G}$:

Используя лемму о подъёме, «поднимем» этот вывод до вывода из S:



$$S = \{\mathrm{P}(\mathrm{x}), \quad \neg\mathrm{P}(\mathbf{f}(\mathrm{x})) \vee \mathrm{R}(\mathrm{x}, \mathbf{g}(\mathrm{x})), \quad \neg\mathrm{R}(\mathrm{x}, \mathrm{u})\},$$
$$\mathcal{G} = \{\mathrm{P}(\mathbf{f}(\mathbf{c})), \quad \neg\mathrm{P}(\mathbf{f}(\mathbf{c})) \vee \mathrm{R}(\mathbf{c}, \mathbf{g}(\mathbf{c})), \quad \neg\mathrm{R}(\mathbf{c}, \mathbf{g}(\mathbf{c}))\}$$

В этой схеме обоснования полноты «*неконструктивен*»
только первый этап: не говорится, как можно эффективно построить
конечное невыполнимое множество $\mathcal{G}$
основных примеров дизъюнктов невыполнимой системы S

Можно представить себе это построение, например, так:

- Будем бесконечно строить (перебирать) **всевозможные** конечные множества $\mathcal{G}$ основных примеров дизъюнктов исходной системы S
- Построив очередное множество $\mathcal{G}$, попытаемся вывести из него $\square$
- Рано или поздно хотя бы из одного построенного $\mathcal{G}$ удастся вывести $\square$ — «поднимем» этот вывод, получив вывод $\square$ из S

А есть ли более эффективный способ вывода $\square$
из произвольной невыполнимой системы дизъюнктов?

Математическая логика и логическое программирование

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

Блок 27

Стратегии резолютивного вывода

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

ВМК МГУ, 2023/2024, осенний семестр

В методе резолюций, как и в методе семантических таблиц, есть свобода выбора (*таблиц; термов; дизъюнктов*), позволяющая строить существенно различающиеся выводы

Например: (*переименование дизъюнктов опущено для наглядности*)

$$S = \left\{ \begin{array}{l} \varphi_1 : \neg \mathrm{Q}(\mathbf{a}, \mathbf{b}) \quad \varphi_2 : \mathrm{R}(\mathbf{a}, \mathbf{b}) \quad \varphi_3 : \mathrm{Q}(\mathrm{x}, \mathrm{x}) \\ \varphi_4 : \neg \mathrm{Q}(\mathrm{x}, \mathrm{y}) \vee \neg \mathrm{Q}(\mathrm{y}, \mathrm{z}) \vee \mathrm{Q}(\mathrm{x}, \mathrm{z}) \\ \varphi_5 : \neg \mathrm{R}(\mathrm{x}, \mathrm{y}) \vee \mathrm{Q}(\mathrm{x}, \mathrm{y}) \end{array} \right\}$$

Вывод из S можно устроить, например, так:

$$\neg \mathrm{Q}(\mathbf{a}, \mathbf{b}) \to \neg \mathrm{R}(\mathbf{a}, \mathbf{b}) \to \square \quad \begin{array}{cc} \varphi_5 & \varphi_2 \\ \downarrow & \downarrow \end{array}$$

$$\neg \mathrm{Q}(\mathbf{a}, \mathbf{b}) \to \neg \mathrm{Q}(\mathbf{a}, \mathrm{y}) \vee \neg \mathrm{Q}(\mathrm{y}, \mathbf{b}) \to \neg \mathrm{Q}(\mathbf{a}, \mathbf{b}) \to \infty \quad \begin{array}{cc} \varphi_4 & \varphi_3 \\ \downarrow & \downarrow \end{array}$$

$$\neg \mathrm{Q}(\mathrm{x}, \mathrm{y}) \vee \neg \mathrm{Q}(\mathrm{y}, \mathrm{z}) \vee \mathrm{Q}(\mathrm{x}, \mathrm{z}) \to \begin{array}{l} \neg \mathrm{Q}(\mathrm{x}, \mathrm{y}) \vee \neg \mathrm{Q}(\mathrm{y}, \mathrm{z}) \vee \\ \neg \mathrm{Q}(\mathrm{x}, \mathrm{y}') \vee \neg \mathrm{Q}(\mathrm{y}', \mathrm{z}) \vee \mathrm{Q}(\mathrm{x}, \mathrm{z}) \end{array} \to \infty \quad \begin{array}{c} \\ \uparrow \\ \varphi_4 \end{array}$$

Стратегии резолютивного вывода

От способа применения правил резолютивного вывода существенно зависит возможность вывести $\square$

Стратегия резолютивного вывода — это набор ограничений на выбор дизъюнктов, к которым применяются правила при построении вывода

Стратегия резолютивного вывода называется полной, если для любой невыполнимой системы дизъюнктов S существует вывод $\square$ из S, построенный согласно этой стратегии

Например, стратегия, никак не ограничивающая выбор дизъюнктов, полна — но существуют и более «эффективные» полные стратегии

Семантическая резолюция

По своему усмотрению выберем интерпретацию $\mathcal{I}$

$\mathcal{I}$-резолюция — это стратегия с одним (следующим) ограничением:

Для дизъюнктов D_1, D_2, к которым применяется правило резолюции, должно быть верно $\mathcal{I} \models D_1$ и $\mathcal{I} \not\models D_2$

Резолютивный вывод, построенный согласно $\mathcal{I}$-резолюции, называется $\mathcal{I}$-резолютивным выводом

Пример: для эрбрановской интерпретации $\mathcal{I} = \emptyset$ и системы

$$S = \{\neg A \vee \neg B \vee C, \quad A \vee C, \quad B \vee C, \quad \neg C\}$$

один из $\mathcal{I}$-резолютивных выводов выглядит так:

$(\mathcal{I} \models D_i)$	$(\mathcal{I} \not\models D_i)$
$D_1 = \neg A \vee \neg B \vee C$	$D_3 = A \vee C$
$D_2 = \neg C$	$D_4 = B \vee C$
$D_1, D_3 \to D_5 = \neg B \vee C \vee C$	$D_2, D_3 \to D_9 = A$
$D_5 \to D_6 = \neg B \vee C$	$D_2, D_4 \to D_{10} = B$
$D_1, D_4 \to D_7 = \neg A \vee C \vee C$	$D_6, D_{10} \to D_{11} = C$
$D_7 \to D_8 = \neg A \vee C$	$D_2, D_{11} \to D_{12} = \square$

Семантическая резолюция

Теорема (о полноте семантической резолюции)
$\mathcal{I}$-резолюция полна для любой интерпретации $\mathcal{I}$

Иными словами, для любой невыполнимой системы дизъюнктов S и любой интерпретации $\mathcal{I}$ существует $\mathcal{I}$-резолютивный вывод $\square$ из S

Доказательство приводить не будем, отметив только, что оно следует предложенной ранее схеме обоснования полноты резолютивного вывода, и что существенные исправления требуется внести только в лемму об основных дизъюнктах

Входная резолюция

Входной резолютивный вывод из системы S,
инициированный дизъюнктом D этой системы, устроен так:

- Нечётные дизъюнкты вывода (при нумерации с единицы) называются центральными, чётные — боковыми
- Первый дизъюнкт вывода — это D
- Все боковые дизъюнкты являются вариантами дизъюнктов из S
- Каждый центральный дизъюнкт, кроме первого, — это резольвента двух предшествующих дизъюнктов

Проще говоря, во входном выводе каждый следующий центральный дизъюнкт — это резольвента предыдущего центрального дизъюнкта и варианта дизъюнкта исходной системы

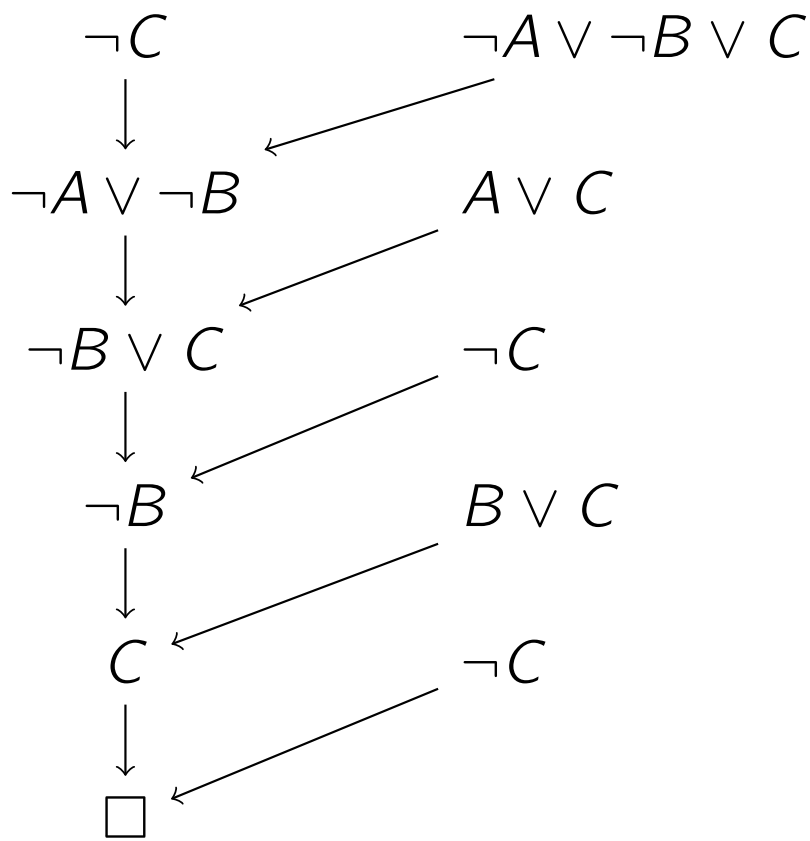
Входная резолюция — это стратегия, согласно которой разрешено строить любые входные резолютивные выводы и только их

Входная резолюция

Пример: один из входных резолютивных выводов из системы

$$S = \{\neg A \vee \neg B \vee C, \quad A \vee C, \quad B \vee C, \quad \neg C\}$$

устроен так:



Входная резолюция

Теорема (о неполноте входной резолюции)

Входная резолюция неполна

Доказательство

Вот пример системы,
из которой не существует входного резолютивного вывода $\square$:

$$S = \{A \vee A, \quad \neg A \vee \neg A\}$$

При построении входного вывода
запрещено применять правило склейки, а значит,
все дизъюнкты входного вывода из S содержат ровно две литеры ▼

Запрет на применение правила склейки —
не единственная причина неполноты входной резолюции

Даже если добавить возможность склеивать дизъюнкт перед
построением резольвенты, то (*как можно показать полным перебором*)
невозможно вывести $\square$, например, из невыполнимой системы

$$\{A \vee C, \quad \neg A \vee C, \quad B \vee \neg C, \quad \neg B \vee \neg C\}$$

Математическая логика и логическое программирование

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

Блок 28

Даша, Саша, Паша, пиво
и метод семантических таблиц
с методом резолюций

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

В блоке 6 была в качестве иллюстрации предложена такая задача

Дано:

- Даша любит Сашу,
$$\varphi_1 = \mathrm{L}(\mathbf{Д}, \mathbf{С})$$
- а Саша любит пиво,
$$\varphi_2 = \mathrm{L}(\mathbf{С}, \mathbf{п})$$
- а Паша любит пиво и всех тех, кто любит то же, что и он
$$\begin{aligned} \varphi_3 &= \mathrm{L}(\mathbf{П}, \mathbf{п}) \\ \psi_1 &= \forall \mathrm{x}\,(\exists \mathrm{y}\,(\mathrm{L}(\mathbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y})) \to \mathrm{L}(\mathbf{П}, \mathrm{x})) \end{aligned}$$

Выяснить, любит ли кто-нибудь Дашу
$$\chi = \exists \mathrm{x}\ \mathrm{L}(\mathrm{x}, \mathbf{Д})$$
$$\varphi_1, \varphi_2, \varphi_3, \psi_1 \models \chi?$$

Или, по теореме о логическом следствии:
$$\models \varphi_1 \,\&\, \varphi_2 \,\&\, \varphi_3 \,\&\, \psi_1 \to \chi?$$

Попробуем решить эту задачу
методом семантических таблиц и методом резолюций

Решение методом семантических таблиц

$\varphi_1 = \mathrm{L}(\textbf{Д}, \textbf{С})$ $\quad$ $\psi_1 = \forall \mathrm{x}\,(\exists \mathrm{y}\,(\mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y})) \to \mathrm{L}(\textbf{П}, \mathrm{x}))$

$\varphi_2 = \mathrm{L}(\textbf{С}, \textbf{п})$ $\quad$ $\psi_2 = \exists \mathrm{y}\,(\mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\textbf{С}, \mathrm{y}))$

$\varphi_3 = \mathrm{L}(\textbf{П}, \textbf{п})$ $\quad$ $\psi_3 = \psi_2 \to \varphi_4$

$\varphi_4 = \mathrm{L}(\textbf{П}, \textbf{С})$

$\chi = \exists \mathrm{x}\,\mathrm{L}(\mathrm{x}, \textbf{Д})$

$$\langle \mid \varphi_1 \,\&\, \varphi_2 \,\&\, \varphi_3 \,\&\, \psi_1 \to \chi \rangle$$

$$\downarrow R{\to},\ L\&,\ L\&,\ L\&$$

$$\langle \varphi_1, \varphi_2, \varphi_3, \psi_1 \mid \chi \rangle$$

$$\downarrow L\forall$$

$$\langle \varphi_1, \varphi_2, \varphi_3, \psi_1, \psi_3 \mid \chi \rangle$$

$$\swarrow \quad L{\to} \quad \searrow$$

$$\langle \varphi_1, \varphi_2, \varphi_3, \varphi_4, \psi_1 \mid \chi \rangle \qquad \langle \varphi_1, \varphi_2, \varphi_3, \psi_1 \mid \chi, \psi_2 \rangle$$

$$\downarrow R\exists$$

$$\langle \varphi_1, \varphi_2, \varphi_3, \psi_1 \mid \chi, \psi_2, \varphi_2 \,\&\, \varphi_3 \rangle$$

$$\swarrow \quad R\& \quad \downarrow$$

$$\langle \varphi_1, \varphi_2, \varphi_3, \psi_1 \mid \chi, \psi_2, \varphi_2 \rangle \qquad \langle \varphi_1, \varphi_2, \varphi_3, \psi_1 \mid \chi, \psi_2, \varphi_3 \rangle$$

Решение методом семантических таблиц

$\varphi_1 = \mathrm{L}(\textbf{Д}, \textbf{С})$ $\quad \psi_1 = \forall \mathrm{x}\,(\exists \mathrm{y}\,(\mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y})) \to \mathrm{L}(\textbf{П}, \mathrm{x}))$

$\varphi_2 = \mathrm{L}(\textbf{С}, \textbf{п})$ $\quad \psi_2 = \exists \mathrm{y}\,(\mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\textbf{С}, \mathrm{y}))$

$\varphi_3 = \mathrm{L}(\textbf{П}, \textbf{п})$ $\quad \psi_3 = \psi_2 \to \varphi_4$

$\varphi_4 = \mathrm{L}(\textbf{П}, \textbf{С})$ $\quad \psi_4 = \exists \mathrm{y}\,(\mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\textbf{Д}, \mathrm{y}))$

$\varphi_5 = \mathrm{L}(\textbf{П}, \textbf{Д})$ $\quad \psi_5 = \psi_4 \to \varphi_5$

$\chi = \exists \mathrm{x}\,\mathrm{L}(\mathrm{x}, \textbf{Д})$

$$\langle \varphi_1, \varphi_2, \varphi_3, \varphi_4, \psi_1 \mid \chi \rangle$$
$$\downarrow L\forall$$
$$\langle \varphi_1, \varphi_2, \varphi_3, \varphi_4, \psi_1, \psi_5 \mid \chi \rangle$$
$$\swarrow \quad L\to \quad \searrow$$
$$\langle \varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5, \psi_1 \mid \chi \rangle \qquad \langle \varphi_1, \varphi_2, \varphi_3, \varphi_4, \psi_1 \mid \chi, \psi_4 \rangle$$
$$\downarrow R\exists \qquad\qquad \downarrow R\exists$$
$$\langle \varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5, \psi_1 \mid \chi, \varphi_5 \rangle \qquad \langle \varphi_1, \varphi_2, \varphi_3, \varphi_4, \psi_1 \mid \chi, \psi_4, \varphi_4 \,\&\, \varphi_1 \rangle$$
$$\swarrow \quad R\& \quad \downarrow$$
$$\langle \varphi_1, \varphi_2, \varphi_3, \varphi_4, \psi_1 \mid \chi, \psi_4, \varphi_1 \rangle \qquad \langle \varphi_1, \varphi_2, \varphi_3, \varphi_4, \psi_1 \mid \chi, \psi_4, \varphi_4 \rangle$$

Получен успешный табличный вывод

Значит, $\varphi_1, \varphi_2, \varphi_3, \psi_1 \models \chi$, то есть кто-то действительно любит Дашу

Решение методом резолюций

$$\varphi_1 = \mathrm{L}(\textbf{Д}, \textbf{С}) \quad \varphi_2 = \mathrm{L}(\textbf{С}, \textbf{п}) \quad \varphi_3 = \mathrm{L}(\textbf{П}, \textbf{п})$$
$$\psi_1 = \forall \mathrm{x}\, (\exists \mathrm{y}\, (\mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y})) \to \mathrm{L}(\textbf{П}, \mathrm{x}))$$
$$\chi = \exists \mathrm{x}\, \mathrm{L}(\mathrm{x}, \textbf{Д})$$
$$\models \varphi_1 \,\&\, \varphi_2 \,\&\, \varphi_3 \,\&\, \psi_1 \to \chi?$$

Этап 1: поставим отрицание над формулой

$$\neg(\varphi_1 \,\&\, \varphi_2 \,\&\, \varphi_3 \,\&\, \psi_1 \to \chi)$$

Этап 2: построим равносильную ПНФ

$$\forall \mathrm{x}\, \forall \mathrm{y}\, \forall \mathrm{z} \left(\begin{array}{c} \mathrm{L}(\textbf{Д}, \textbf{С}) \,\&\, \mathrm{L}(\textbf{С}, \textbf{п}) \,\&\, \mathrm{L}(\textbf{П}, \textbf{п}) \\ \&(\neg \mathrm{L}(\textbf{П}, \mathrm{y}) \vee \neg \mathrm{L}(\mathrm{x}, \mathrm{y}) \vee \mathrm{L}(\textbf{П}, \mathrm{x})) \\ \& \neg \mathrm{L}(\mathrm{z}, \textbf{Д}) \end{array} \right)$$

Этап 3: построим равновыполнимую ССФ

Формула выше — ССФ

Этап 4: перейдём к системе дизъюнктов

$$S = \left\{ \begin{array}{ccc} \mathrm{L}(\textbf{Д}, \textbf{С}), & \mathrm{L}(\textbf{С}, \textbf{п}), & \mathrm{L}(\textbf{П}, \textbf{п}), \\ \multicolumn{3}{c}{\neg \mathrm{L}(\textbf{П}, \mathrm{y}) \vee \neg \mathrm{L}(\mathrm{x}, \mathrm{y}) \vee \mathrm{L}(\textbf{П}, \mathrm{x}),} \\ \multicolumn{3}{c}{\neg \mathrm{L}(\mathrm{z}, \textbf{Д})} \end{array} \right\}$$

Решение методом резолюций

Этап 5: попробуем вывести пустой дизъюнкт

$$\left\{\begin{array}{c} \mathrm{L}(\textbf{Д}, \textbf{С}), \quad \mathrm{L}(\textbf{С}, \textbf{п}), \quad \mathrm{L}(\textbf{П}, \textbf{п}), \\ \neg\mathrm{L}(\textbf{П}, \mathrm{y}) \vee \neg\mathrm{L}(\mathrm{x}, \mathrm{y}) \vee \mathrm{L}(\textbf{П}, \mathrm{x}), \\ \neg\mathrm{L}(\mathrm{z}, \textbf{Д}) \end{array}\right\}$$

$$\begin{array}{ccccccccc} & \neg\mathrm{L}(\textbf{П}, \mathrm{y}') \vee \neg\mathrm{L}(\mathrm{x}', \mathrm{y}') \vee \mathrm{L}(\textbf{П}, \mathrm{x}') & & \neg\mathrm{L}(\textbf{П}, \mathrm{y}') \vee \neg\mathrm{L}(\mathrm{x}', \mathrm{y}') \vee \mathrm{L}(\textbf{П}, \mathrm{x}') & & \mathrm{L}(\textbf{С}, \textbf{п}) & \\ & \{\mathrm{z}/\textbf{П}, \mathrm{x}'/\textbf{Д}, \mathrm{y}'/\mathrm{y}\} \downarrow & & \{\mathrm{x}'/\textbf{С}, \mathrm{y}'/\mathrm{y}\} \downarrow & & \varepsilon \downarrow & \\ \neg\mathrm{L}(\mathrm{z}, \textbf{Д}) & \to \neg\mathrm{L}(\textbf{П}, \mathrm{y}) \vee \neg\mathrm{L}(\textbf{Д}, \mathrm{y}) & \to & \neg\mathrm{L}(\textbf{П}, \textbf{С}) & \to \neg\mathrm{L}(\textbf{П}, \mathrm{y}) \vee \neg\mathrm{L}(\textbf{С}, \mathrm{y}) & \to & \neg\mathrm{L}(\textbf{С}, \textbf{п}) & \to \square \\ & & & \{\mathrm{y}/\textbf{С}\} \uparrow & & \{\mathrm{y}/\textbf{п}\} \uparrow & & \\ & & & \mathrm{L}(\textbf{Д}, \textbf{С}) & & \mathrm{L}(\textbf{П}, \textbf{п}) & & \end{array}$$

Это успешный входной резолютивный вывод $\square$,
инициированный дизъюнктом $\neg\mathrm{L}(\mathrm{z}, \textbf{Д})$

Значит, $\varphi_1, \varphi_2, \varphi_3, \psi_1 \models \chi$, то есть кто-то действительно любит Дашу

А кто?

Решение методом резолюций

$$
\begin{array}{ll}
\neg \mathrm{L}(\mathrm{z}, \textbf{Д}) & \\
\downarrow & \theta_1 = \{\mathrm{z}/\textbf{П}, \mathrm{x}'/\textbf{Д}, \mathrm{y}'/\mathrm{y}\} \\
\neg \mathrm{L}(\textbf{П}, \mathrm{y}) \vee \neg \mathrm{L}(\textbf{Д}, \mathrm{y}) & \\
\downarrow & \theta_2 = \{\mathrm{y}/\textbf{С}\} \\
\neg \mathrm{L}(\textbf{П}, \textbf{С}) & \\
\downarrow & \theta_3 = \{\mathrm{x}'/\textbf{С}, \mathrm{y}'/\mathrm{y}\} \\
\neg \mathrm{L}(\textbf{П}, \mathrm{y}) \vee \neg \mathrm{L}(\textbf{С}, \mathrm{y}) & \\
\downarrow & \theta_4 = \{\mathrm{y}/\textbf{п}\} \\
\neg \mathrm{L}(\textbf{С}, \textbf{п}) & \\
\downarrow & \theta_5 = \varepsilon \\
\square &
\end{array}
$$

Тайный поклонник Даши в выводе обозначен переменной z

Посмотрим, как эта переменная изменялась унификаторами:

$$\mathrm{z}\theta_1\theta_2\theta_3\theta_4\theta_5 = \textbf{П}$$

Оказывается, что Дашу любит Паша

(*могут быть и другие поклонники, но про них мы ничего не знаем*)

А что это за «трюк» с применением подстановок,

в каких случаях и как именно он работает?

Заключительный пример

Перед детальным обсуждением этого «трюка» —
пример задачи, для которой так вычислить ответ нельзя

- Если вечером будет дождь, то мы пойдём в кино
$$\text{Будет}(\textbf{дождь}) \to \text{Досуг}(\textbf{кино})$$
- Если вечером дождя не будет, то мы пойдём гулять в парк
$$\neg\text{Будет}(\textbf{дождь}) \to \text{Досуг}(\textbf{парк})$$

Где мы проведём этот вечер?

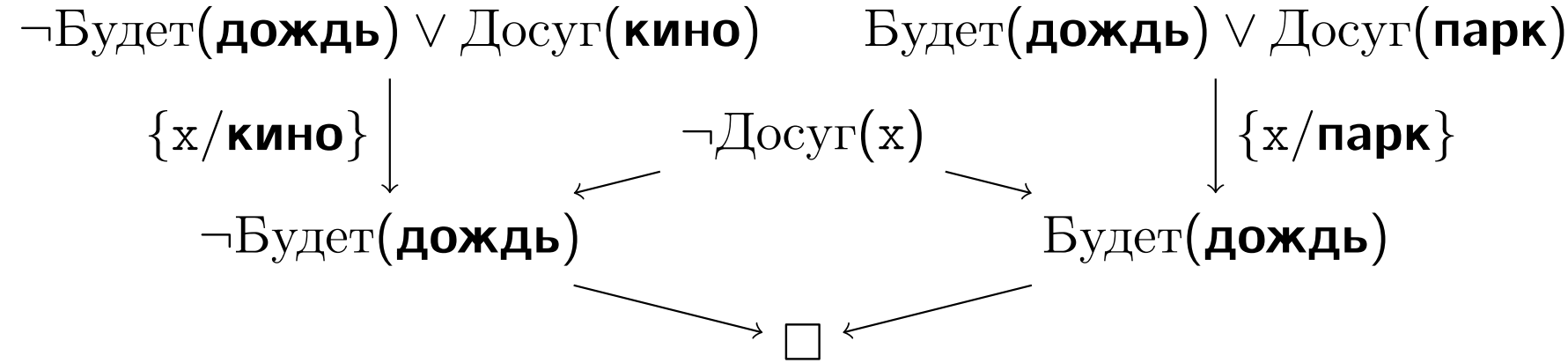
Действуя по аналогии с задачей о Даше, Саше, Паше и пиве, вопрос можно ослабить так, понадеявшись на тот же «трюк» с вычислением:
Есть ли такое место, которое мы посетим этим вечером?
$$\exists x\ \text{Досуг}(x)\ ?$$

Заключительный пример

По аналогии с задачей о Даше, Саше, Паше и пиве получим систему дизъюнктов:

$$\left\{ \begin{array}{l} \neg\text{Будет}(\textbf{дождь}) \vee \text{Досуг}(\textbf{кино}) \\ \text{Будет}(\textbf{дождь}) \vee \text{Досуг}(\textbf{парк}) \\ \neg\text{Досуг}(\text{x}) \end{array} \right\}$$

Такое место действительно существует:



Но по выводу невозможно понять, куда же мы пойдём

Невозможно определить место досуга достоверно и однозначно, пока не стало известно, будет ли вечером дождь

Математическая логика и логическое программирование

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

Блок 29

Хорновские дизъюнкты

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

ВМК МГУ, 2023/2024, осенний семестр

Правилом будем называть дизъюнкт, содержащий ровно одну положительную литеру, то есть имеющий вид

$$\neg A_1 \vee \cdots \vee \neg A_k \vee B$$

Запросом будем называть дизъюнкт, не содержащий ни одной положительной литеры (в том числе $\square$), то есть имеющий вид

$$\neg A_1 \vee \cdots \vee \neg A_k$$

Хорновскими дизъюнктами называются правила и запросы

Утверждение. $\neg A_1 \vee \cdots \vee \neg A_k \vee B \sim A_1 \,\&\, \ldots \,\&\, A_k \to B$

Утверждение. $\neg A_1 \vee \cdots \vee \neg A_k \sim \neg(A_1 \,\&\, \ldots \,\&\, A_k)$

Основываясь на этих утверждениях, дизъюнкты $\neg A_1 \vee \cdots \vee \neg A_k \vee B$ и $\neg C_1 \vee \cdots \vee \neg C_k$ будем иногда записывать как формулы $A_1 \,\&\, \ldots \,\&\, A_k \to B$ и $\neg(C_1 \,\&\, \ldots \,\&\, C_k)$

Утверждение. Контрарную пару с заданной литерой запроса может образовывать не более одной литеры правила

Утверждение. Ни к какой паре запросов нельзя применить правило резолюции

Утверждение. Резольвента запроса и правила является запросом

Правило $A_1 \,\&\, \ldots \,\&\, A_k \to B$ — это естественный способ представления причинно-следственных взаимосвязей:

- Если справедливы факты $A_1, \ldots, A_k$, то справедлив и факт B
  - То есть $A_1, \ldots, A_k$ — достаточное условие справедливости B
  - Если $k = 0$, то просто «Справедлив факт B»
- Чтобы решить задачу, записанную в виде B, достаточно решить задачи, записанные в виде $A_1, \ldots, A_k$, и совместить ответы
  - Если $k = 0$, то «Решение задачи B очевидно»

Формула $A_1 \,\&\, \ldots \,\&\, A_k$, отрицанием которой является запрос, — это естественный способ представления вопроса к задаче:

- Требуется проверить справедливость набора (*взаимосвязанных*) фактов $A_1, \ldots, A_k$
- Требуется решить набор задач, записанных в виде $A_1, \ldots, A_k$

Например, система дизъюнктов

$$\left\{ \begin{array}{c} \mathrm{L}(\textbf{Д}, \textbf{С}), \qquad \mathrm{L}(\textbf{С}, \textbf{п}), \qquad \mathrm{L}(\textbf{П}, \textbf{п}), \\ \mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y}) \to \mathrm{L}(\textbf{П}, \mathrm{x}), \\ \neg \mathrm{L}(\mathrm{z}, \textbf{Д}) \end{array} \right\}$$

из задачи о Даше, Саше, Паше и пиве — это система хорновских дизъюнктов:

- В части «Дано» записаны правила:

$$\mathrm{L}(\textbf{Д}, \textbf{С})$$
$$\mathrm{L}(\textbf{С}, \textbf{п})$$
$$\mathrm{L}(\textbf{П}, \textbf{п})$$
$$\mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y}) \to \mathrm{L}(\textbf{П}, \mathrm{x})$$

- Вопрос задачи, записанный в виде формулы и дополненный отрицанием в результате преобразования к дизъюнктам, — это запрос:

$$\neg \mathrm{L}(\mathrm{z}, \textbf{Д})$$

Математическая логика и логическое программирование

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

Блок 30

Вычислительные возможности метода резолюций

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

ВМК МГУ, 2023/2024, осенний семестр

Теорема о входной резолюции как средстве вычисления

Пусть

- S **— система дизъюнктов-правил,**
- Q_1 **— дизъюнкт-запрос,**
- $Q_1, D_1, Q_2, D_2, \ldots, Q_k, D_k, \square$**,** $k \geq 0$**, — успешный входной резолютивный вывод из** $S \cup \{Q_1\}$**,**
- $\theta_1, \ldots, \theta_k$ **— наиболее общие унификаторы, согласно которым в выводе строятся резольвенты** $Q_2, \ldots, Q_k, \square$ **соответственно, и**
- $\mathrm{Var}_{Q_1\theta_1\ldots\theta_k} = \{\mathtt{x}_1, \ldots, \mathtt{x}_n\}$**.**

Тогда $S \models \forall\widetilde{\mathtt{x}^{\mathtt{n}}}\ ((\neg Q_1)\theta_1 \ldots \theta_k)$

Доказательство (индукцией по k).

База: $k = 0$

Тогда $Q_1 = \square$, $\mathrm{Var}_{Q_1} = \emptyset$ и $S \not\models \square$, а значит, $S \models \neg Q_1$

Теорема о входной резолюции как средстве вычисления

Доказательство (индуктивный переход).

$$(Q_1, D_1 \xrightarrow{\theta_1} Q_2, D_2 \xrightarrow{\theta_2} \ldots \xrightarrow{\theta_{k-1}} Q_k, D_k \xrightarrow{\theta_k} \square)$$

Пусть утверждение справедливо для $k < N$ для заданного N, $N \geq 1$; покажем, что оно справедливо и для $k = N$

Далее $\forall\ldots\ \varphi$ означает $\forall\widetilde{\mathrm{x}^{\mathrm{n}}}\ \varphi$, где $\mathrm{Var}_\varphi = \{\mathrm{x}_1, \ldots, \mathrm{x}_n\}$

Пусть $Q_1 = \neg(A_1 \,\&\, \ldots \,\&\, A_q \,\&\, B)$, $Q_2 = \neg(A_1 \,\&\, \ldots \,\&\, A_p)\theta_1$, $D_1 = A_{q+1} \,\&\, \ldots \,\&\, A_p \to C$, $B\theta_1 = C\theta_1$ и $\eta = \theta_2 \ldots \theta_k$

По индуктивному предположению, верно $S \models \forall\ldots\ ((\neg Q_2)\eta)$

Значит, верно $S \models \forall\ldots\ ((A_1 \,\&\, \ldots \,\&\, A_q)\theta_1\eta)$ и $S \models \forall\ldots\ ((A_{q+1} \,\&\, \ldots \,\&\, A_p)\theta_1\eta)$ (почему?)

При этом $S \models \forall\ldots\ (A_{q+1} \,\&\, \ldots \,\&\, A_p \to C)$, (почему?)
а значит, и $S \models \forall\ldots\ ((A_{q+1} \,\&\, \ldots \,\&\, A_p \to C)\theta_1\eta)$ (почему?)

Тогда, $S \models \forall\ldots\ (C\theta_1\eta)$ (почему?)

При этом $C\theta_1 = B\theta_1$, а значит, $S \models \forall\ldots\ (B\theta_1\eta)$

Следовательно, $S \models \forall\ldots\ ((\neg Q_1)\theta_1\eta)$ ▼

Вычисление при помощи входной резолюции (примеры)

Пример: Даша, Саша, Паша и пиво

$$S = \left\{ \begin{array}{c} \mathrm{L}(\textbf{Д}, \textbf{С}), \qquad \mathrm{L}(\textbf{С}, \textbf{п}), \qquad \mathrm{L}(\textbf{П}, \textbf{п}), \\ \mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y}) \to \mathrm{L}(\textbf{П}, \mathrm{x}) \end{array} \right\} \qquad Q_1 = \neg \mathrm{L}(\mathrm{z}, \textbf{Д})$$

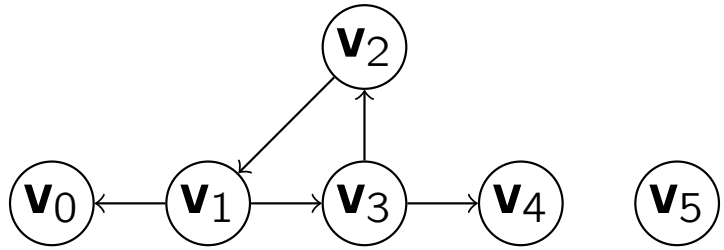
$$\begin{array}{rcll} & & \neg \mathrm{L}(\mathrm{z}, \textbf{Д}) & \\ & & \downarrow & \theta_1 = \{\mathrm{z}/\textbf{П}, \mathrm{x}'/\textbf{Д}, \mathrm{y}'/\mathrm{y}\} \\ \mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y}) \to \mathrm{L}(\textbf{П}, \mathrm{x}) & \longrightarrow & \neg(\mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\textbf{Д}, \mathrm{y})) & \\ & & \downarrow & \theta_2 = \{\mathrm{y}/\textbf{С}\} \\ \mathrm{L}(\textbf{Д}, \textbf{С}) & \longrightarrow & \neg \mathrm{L}(\textbf{П}, \textbf{С}) & \\ & & \downarrow & \theta_3 = \{\mathrm{x}'/\textbf{С}, \mathrm{y}'/\mathrm{y}\} \\ \mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\mathrm{x}, \mathrm{y}) \to \mathrm{L}(\textbf{П}, \mathrm{x}) & \longrightarrow & \neg(\mathrm{L}(\textbf{П}, \mathrm{y}) \,\&\, \mathrm{L}(\textbf{С}, \mathrm{y})) & \\ & & \downarrow & \theta_4 = \{\mathrm{y}/\textbf{п}\} \\ \mathrm{L}(\textbf{П}, \textbf{п}) & \longrightarrow & \neg \mathrm{L}(\textbf{С}, \textbf{п}) & \\ & & \downarrow & \theta_5 = \varepsilon \\ \mathrm{L}(\textbf{С}, \textbf{п}) & \longrightarrow & \square & \end{array}$$

По только что доказанной теореме, верно $S \models \mathrm{L}(\mathrm{z}, \textbf{Д})\theta_1\theta_2\theta_3\theta_4\theta_5$

То есть $S \models \mathrm{L}(\textbf{П}, \textbf{Д})$: Паша действительно любит Дашу

Вычисление при помощи входной резолюции (примеры)

Пример: проверка достижимости в графе



Достижима ли вершина $\mathbf{v}_4$ из $\mathbf{v}_1$, и если да, то по какому пути?

Попробуем записать эту задачу на языке логики предикатов и решить при помощи входной резолюции

Вершины графа $(\mathbf{v}_0, \mathbf{v}_1, \dots)$ — это константы

Ещё так получится, что вершинами графа можно считать все термы, но в рассматриваемом примере это неважно

Тот факт, что (u, w) — дуга графа, будет записываться в виде атома $\mathrm{Arc}(u, w)$

Тогда в нашем распоряжении есть пять фактов, отвечающих пяти дугам:

$$\mathrm{Arc}(\mathbf{v}_1, \mathbf{v}_0) \quad \mathrm{Arc}(\mathbf{v}_1, \mathbf{v}_3) \quad \mathrm{Arc}(\mathbf{v}_3, \mathbf{v}_2)$$
$$\mathrm{Arc}(\mathbf{v}_2, \mathbf{v}_1) \quad \mathrm{Arc}(\mathbf{v}_3, \mathbf{v}_4)$$

Вычисление при помощи входной резолюции (примеры)

Пример: проверка достижимости в графе

Путь $u_1 \to u_2 \to \cdots \to u_k$ будет записываться так (*чтобы не путать дуги и импликации*):

$$u_1 \hookrightarrow u_2 \hookrightarrow \ldots \hookrightarrow u_k$$

$\hookrightarrow$ — это функциональный символ местности 2

Символ $\hookrightarrow$ положим ассоциативным вправо: $x \hookrightarrow y \hookrightarrow z = x \hookrightarrow (y \hookrightarrow z)$

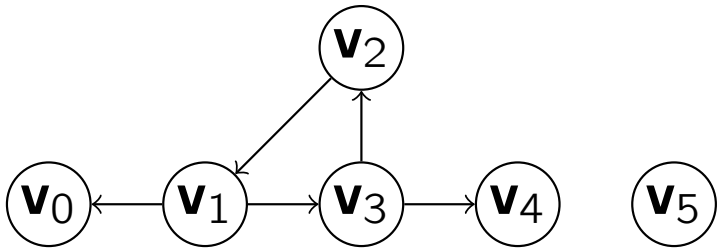
Тот факт, что вершина w достижима из u по пути π, будем записывать в виде атома $\mathrm{Reach}(u, w, \pi)$

Тогда для такого трёхместного отношения достижимости справедливы следующие свойства (*индуктивное определение*):

- Любая вершина достижима из себя по тривиальному пути
$$\forall \mathrm{x}\ \mathrm{Reach}(\mathrm{x}, \mathrm{x}, \mathrm{x})$$
- Если в графе есть дуга $\mathrm{x} \hookrightarrow \mathrm{y}$ и из y по пути π достижима вершина z, то из x по пути $\mathrm{x} \hookrightarrow \pi$ достижима вершина z
$$\forall \mathrm{x}\ \forall \mathrm{y}\ \forall \mathrm{z}\ \forall \mathrm{p}\ (\mathrm{Arc}(\mathrm{x}, \mathrm{y}) \,\&\, \mathrm{Reach}(\mathrm{y}, \mathrm{z}, \mathrm{p}) \to \mathrm{Reach}(\mathrm{x}, \mathrm{z}, \mathrm{x} \hookrightarrow \mathrm{p}))$$

Вычисление при помощи входной резолюции (примеры)

Пример: проверка достижимости в графе



$\mathbf{v}_1 \rightsquigarrow \mathbf{v}_4$?

Входные данные задачи можно записать в виде системы дизъюнктов:

$$S = \left\{ \begin{array}{lll} \mathrm{Arc}(\mathbf{v}_1, \mathbf{v}_0) & \mathrm{Arc}(\mathbf{v}_1, \mathbf{v}_3) & \mathrm{Arc}(\mathbf{v}_3, \mathbf{v}_2) \\ \mathrm{Arc}(\mathbf{v}_2, \mathbf{v}_1) & \mathrm{Arc}(\mathbf{v}_3, \mathbf{v}_4) & \mathrm{Reach}(\mathrm{x}, \mathrm{x}, \mathrm{x}) \\ \multicolumn{3}{l}{\mathrm{Arc}(\mathrm{x}, \mathrm{y}) \,\&\, \mathrm{Reach}(\mathrm{y}, \mathrm{z}, \mathrm{p}) \to \mathrm{Reach}(\mathrm{x}, \mathrm{z}, \mathrm{x} \hookrightarrow \mathrm{p})} \end{array} \right\}$$

Вопрос к задаче можно записать в виде формулы $\exists \mathrm{p}\, \mathrm{Reach}(\mathbf{v}_1, \mathbf{v}_4, \mathrm{p})$

Отрицание этой формулы — это дизъюнкт-запрос

$$\varphi = \neg \mathrm{Reach}(\mathbf{v}_1, \mathbf{v}_4, \mathrm{p})$$

Чтобы убедиться, что вершина $\mathbf{v}_4$ достижима из $\mathbf{v}_1$, и найти соответствующий путь, достаточно построить успешный входной вывод $\square$ из $S \cup \{\varphi\}$, инициированный φ, и применить недавно доказанную теорему

Вычисление при помощи входной резолюции (примеры)

Пример: проверка достижимости в графе

$$S = \left\{ \begin{array}{lll} \mathrm{Arc}(\mathbf{v}_1, \mathbf{v}_0) & \mathrm{Arc}(\mathbf{v}_1, \mathbf{v}_3) & \mathrm{Arc}(\mathbf{v}_3, \mathbf{v}_2) \\ \mathrm{Arc}(\mathbf{v}_2, \mathbf{v}_1) & \mathrm{Arc}(\mathbf{v}_3, \mathbf{v}_4) & \mathrm{Reach}(\mathrm{x}, \mathrm{x}, \mathrm{x}) \\ \multicolumn{3}{l}{\mathrm{Arc}(\mathrm{x}, \mathrm{y}) \,\&\, \mathrm{Reach}(\mathrm{y}, \mathrm{z}, \mathrm{p}) \to \mathrm{Reach}(\mathrm{x}, \mathrm{z}, \mathrm{x} \hookrightarrow \mathrm{p})} \end{array} \right\}$$

$$\neg\mathrm{Reach}(\mathbf{v}_1, \mathbf{v}_4, \mathrm{p})$$

$$\theta_1 = \{\mathrm{x}/\mathbf{v}_1, \mathrm{z}/\mathbf{v}_4, \mathrm{p}/\mathbf{v}_1 \hookrightarrow \mathrm{p}'\} \qquad \mathrm{Arc}(\mathrm{x}, \mathrm{y}) \,\&\, \mathrm{Reach}(\mathrm{y}, \mathrm{z}, \mathrm{p}') \to \mathrm{Reach}(\mathrm{x}, \mathrm{z}, \mathrm{x} \hookrightarrow \mathrm{p}')$$

$$\neg(\mathrm{Arc}(\mathbf{v}_1, \mathrm{y}) \,\&\, \mathrm{Reach}(\mathrm{y}, \mathbf{v}_4, \mathrm{p}'))$$

$$\theta_2 = \{\mathrm{y}/\mathbf{v}_3\}\{\mathrm{p}'/\mathrm{p}, \mathrm{p}'/\mathrm{p}\} \qquad \mathrm{Arc}(\mathbf{v}_1, \mathbf{v}_3)$$

$$\neg\mathrm{Reach}(\mathbf{v}_3, \mathbf{v}_4, \mathrm{p})$$

$$\theta_3 = \{\mathrm{x}/\mathbf{v}_3, \mathrm{z}/\mathbf{v}_4, \mathrm{p}/\mathbf{v}_3 \hookrightarrow \mathrm{p}'\} \qquad \mathrm{Arc}(\mathrm{x}, \mathrm{y}) \,\&\, \mathrm{Reach}(\mathrm{y}, \mathrm{z}, \mathrm{p}') \to \mathrm{Reach}(\mathrm{x}, \mathrm{z}, \mathrm{x} \hookrightarrow \mathrm{p}')$$

$$\neg(\mathrm{Arc}(\mathbf{v}_3, \mathrm{y}) \,\&\, \mathrm{Reach}(\mathrm{y}, \mathbf{v}_4, \mathrm{p}'))$$

$$\theta_4 = \{\mathrm{y}/\mathbf{v}_4\}\{\mathrm{p}'/\mathrm{p}, \mathrm{p}/\mathrm{p}'\} \qquad \mathrm{Arc}(\mathbf{v}_3, \mathbf{v}_4)$$

$$\neg\mathrm{Reach}(\mathbf{v}_4, \mathbf{v}_4, \mathrm{p})$$

$$\theta_5 = \{\mathrm{p}/\mathbf{v}_4, \mathrm{x}/\mathbf{v}_4\} \qquad \mathrm{Reach}(\mathrm{x}, \mathrm{x}, \mathrm{x})$$

$$\square$$

$$\mathrm{Reach}(\mathbf{v}_1, \mathbf{v}_4, \mathrm{p})\theta_1\theta_2\theta_3\theta_4\theta_5 = \mathrm{Reach}(\mathbf{v}_1, \mathbf{v}_4, \mathbf{v}_1 \hookrightarrow \mathbf{v}_3 \hookrightarrow \mathbf{v}_4)$$

Следовательно, вершина $\mathbf{v}_4$ достижима из $\mathbf{v}_1$ по пути $\mathbf{v}_1 \hookrightarrow \mathbf{v}_3 \hookrightarrow \mathbf{v}_4$

Математическая логика и логическое программирование

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

Блок 31

Хорновские логические программы:
синтаксис,
декларативная семантика,
правильные ответы

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

ВМК МГУ, 2023/2024, осенний семестр

Несколько слов о парадигмах программирования

Две основные парадигмы программирования:

1. Императивная
   - Программа — это набор команд (инструкций)
   - Семантика программы задаётся как способ пошагового (последовательного) выполнения команд
   - На каждом шаге выполнения текущей командой преобразуются текущие значения данных и определяется то, какая команда должна быть выполнена следующей
2. Декларативная
   - Программа — это набор свойств, задающих (определяющих) правильный результат выполнения
   - В основной семантике программы не используются понятия выполнения, промежуточных значений данных, текущей команды и т.п., определяется только общий вид желаемого результата
   - В декларативно разработанной программе, запускающейся на «обычном» компьютере, в связи с пошаговой природой компьютера и вычислений используется вспомогательная семантика: императивная, согласующаяся с основной декларативной

Несколько слов о парадигмах программирования

Пример

Задача: вскипятить воду в электрическом чайнике

Решение в императивной парадигме (*упрощённое*):

- Если в чайнике нет воды, то налить её
- Нажать на кнопку кипячения
- Дождаться, когда кнопка кипячения отожмётся

Решение в декларативной парадигме (*где-то упрощённое, где-то усложнённое для наглядной аналогии*):

- Желаемый результат — это чайник, в котором горячая вода
- Вода появляется в чайнике, если её налить
- Кнопка кипячения может отжаться только если перед этим нажата
- Нажатая кнопка кипячения отжимается $\Leftrightarrow$ вода в чайнике горячая

Несколько слов о парадигмах программирования

Императивная парадигма программирования — самая популярная и известная, так как эта парадигма

- используется на «низком уровне» в современных вычислительных устройствах и при этом
- достаточно понятна и естественна для программиста и
- обычно изучается первой (и иногда единственной)

Современные языки программирования, как правило, мультипарадигменны (принадлежат нескольким парадигмам), но зачастую можно определить *преобладающую* парадигму языка

Примеры языков, в которых преобладает императивная парадигма: C, C++, Pascal, Java, Python (*хотя местами можно и поспорить о том, что в нём преобладает*), Perl, PHP, машинные коды и ассемблерные языки, ..., ..., ...

Основная математическая вычислительная модель этой парадигмы: машина Тьюринга

Несколько слов о парадигмах программирования

Две самых известных декларативных парадигмы:

1. Функциональная
2. Логическая

В функциональной парадигме:

- Программа — это функция (в математическом понимании), записанная при помощи базового набора функций и операций композиции функций
- В основной семантике не говорится, **как** вычислить функцию программы, говорится только **что** это за функция

Несколько известных языков программирования с преобладающей функциональной парадигмой: Lisp/Scheme, Erlang, Scala (см. JVM), Haskell, ML (см. OCaml), Python (*хотя он по большей части императивный, но и в функциональном смысле тоже применяется*), ...

Основная математическая вычислительная модель этой парадигмы: λ-исчисление

Несколько слов о парадигмах программирования

В логической парадигме:

- Программа — это набор логических формул, представляющих собой определение результата, совокупность формул, представляющих собой критерий результата
- Правильный результат — это логическое следствие программы как набора формул
- В основной семантике не говорится, **как** вычислять (извлекать) логические следствия, говорится только **что** является основанием для извлечения следствий

Несколько известных языков программирования с преобладающей логической парадигмой: **Prolog**, Datalog, Planner, Mercury, Gödel, ...

Основная математическая вычислительная модель этой парадигмы: логический вывод (логические исчисления)

ХЛП: синтаксис

Хорновская логическая программа (ХЛП) *сигнатуры σ логики предикатов* — это конечная последовательность программных утверждений, каждое из которых представляет собой факт или правило

Факт имеет вид «$A;$», где A — атом логики предикатов

Правило имеет вид «$A \leftarrow B_1, \ldots, B_k;$», где:

- $k \geq 1$
- A — заголовок: атом логики предикатов
- $B_1, \ldots, B_k$ — тело: последовательность атомов логики предикатов, разделённых запятой

Запрос к ХЛП (или, по-другому, целевое утверждение, или просто цель) имеет вид «$?C_1, \ldots, C_k$», где

- $k \geq 0$, и для случая $k = 0$ запрос принято записывать так: $\square$
- $C_1, \ldots, C_k$ — тело

Чтобы избежать путаницы, далее правилами и запросами будем называть элементы ХЛП, а формулы — дизъюнктами-правилами и дизъюнктами-запросами

ХЛП: синтаксис

Иными словами, ХЛП и запрос к ней задаются следующей БНФ:

ХЛП	::=	| *утверждение ХЛП*
утверждение	::=	*факт* | *правило*
факт	::=	*атом*;
правило	::=	*заголовок* $\leftarrow$ *тело*;
заголовок	::=	*атом*
тело	::=	*атом* | *атом*, *тело*
запрос	::=	$\square$ | ? *тело*

Для технического единообразия будем считать, что факт — это правило с пустым телом:

$$\text{«}A;\text{»} = \text{«}A \leftarrow;\text{»}$$

Каждый атом в теле запроса (цели) принято также называть подцелью

Переменные, содержащиеся в запросе (цели), принято называть целевыми

ХЛП: синтаксис

Если захотите транслировать ХЛП и запрос в язык Prolog, то для этого достаточно сделать следующее:

1. Заменить все «;» на «.», «←» на «:-», «?» на «?-» и добавить «.» в конце запроса
   - В курсе оставим синтаксис ХЛП как есть, чтобы не изменять сложившуюся математику ради одного конкретного языка программирования

2. Начинать все переменные с прописной (большой) буквы, а остальные идентификаторы — со строчной (маленькой)
   - Будем стараться придерживаться такого написания в курсе, дополнительно различая категории символов шрифтами: $\mathtt{X}$ — переменная, **a** — константа или функциональный символ, p — предикатный символ

ХЛП: синтаксис

Примеры

Правило: $\text{любит}(\textbf{паша}, \texttt{Y}) \leftarrow \text{любит}(\texttt{Y}, \texttt{X}), \text{любит}(\textbf{паша}, \texttt{X});$

- Заголовок: «$\text{любит}(\textbf{паша}, \texttt{Y})$»
- Тело: «$\text{любит}(\texttt{Y}, \texttt{X}), \text{любит}(\textbf{паша}, \texttt{X})$»
- Переменные: $\texttt{Y}$, $\texttt{X}$
- Константы: **паша**
- Предикатные символы: любит

Факт: $\text{любит}(\textbf{паша}, \textbf{пиво});$ $\text{любит}(\textbf{паша}, \textbf{пиво}) \leftarrow;$

Запрос: $?\text{умный}(\texttt{X}), \text{добрый}(\texttt{X}), \text{красивый}(\texttt{X}), \text{любит}(\texttt{X}, \textbf{даша})$

- В запросе содержатся 4 подцели: «$\text{умный}(\texttt{X})$», «$\text{добрый}(\texttt{X})$», «$\text{красивый}(\texttt{X})$», «$\text{любит}(\texttt{X}, \textbf{даша})$»
- $\texttt{X}$ — целевая переменная

ХЛП: декларативная семантика

Ряду элементов ξ синтаксиса ХЛП можно естественным образом сопоставить формулу логики предикатов Φ_ξ:

Элемент	общий вид ξ	формула Φ_ξ
Факт	$A;$	$\forall\ldots\, A$
Тело	$B_1, \ldots, B_k$	$B_1 \,\&\, \ldots \,\&\, B_k$
Правило	$A \leftarrow \beta$	$\forall\ldots\, (\Phi_\beta \to A)$
Запрос	$?\gamma$	Φ_γ

Здесь, как и в блоке 30, $\forall\ldots\, \varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})$ означает $\forall\widetilde{\mathrm{x}}^{\mathrm{n}}\ \varphi(\widetilde{\mathrm{x}}^{\mathrm{n}})$

Хорновской логической программе $\mathcal{P} = (\mathcal{R}_1 \ldots \mathcal{R}_m)$ сопоста́вим систему формул $S_\mathcal{P} = \{\Phi_{\mathcal{R}_1}, \ldots, \Phi_{\mathcal{R}_m}\}$

ХЛП: декларативная семантика

Содержательно, декларативная (основная) семантика ХЛП $\mathcal{P}$ и запроса $\mathcal{Q}$ к ней устроена так:

- Программа — это база имеющихся заведомо верных знаний
  - Правило «$A \leftarrow [B_1, \ldots, B_k];$» — это утверждение о том, что для любых значений переменных утверждение A верно [если для тех же значений верны все утверждения $B_1, \ldots, B_k$]
- Запрос к программе — это входные данные, определяющие вопрос об имеющихся знаниях, на который требуется ответить, записав ответ в целевые переменные (*как запрос к базе данных*)
  - Запрос «$?C_1, \ldots, C_m$» отвечает вопросу «для каких значений целевых переменных становятся одновременно верными все утверждения $C_1, \ldots, C_m$?»
- Правильный ответ на запрос к программе — это значения целевых переменных запроса, при которых этот запрос как формула следует из программы как формулы

ХЛП: декларативная семантика

Формально, центральное понятие декларативной семантики — это правильный ответ, и это понятие определяется так

$\mathrm{Var}_{\mathcal{Q}} = \mathrm{Var}_{\Phi_{\mathcal{Q}}}$ — множество всех переменных запроса $\mathcal{Q}$

Ответ на запрос $\mathcal{Q}$ — это подстановка θ, такая что $\mathrm{Dom}_\theta \subseteq \mathrm{Var}_{\mathcal{Q}}$

Правильный ответ на запрос $\mathcal{Q}$ к программе $\mathcal{P}$ — это ответ θ на запрос $\mathcal{Q}$, для которого выполнено соотношение

$$S_{\mathcal{P}} \models \forall \ldots (\Phi_{\mathcal{Q}}\theta)$$

ХЛП: декларативная семантика

Пример

Программа $\mathcal{P}$:

пернатый(**орёл**); есть_живой(**орёл**); летает(**орёл**);
пернатый(**чайка**); есть_живой(**чайка**); летает(**чайка**);
пернатый(**пингвин**); есть_живой(**пингвин**);
пернатый(**велоцираптор**);
птица(X) ← пернатый(X), есть_живой(X);

Запрос $\mathcal{Q}$:

$$?\text{птица}(\texttt{X}), \text{летает}(\texttt{X})$$

В фактах перечислено всё, что известно про пернатость, современность и способность летать четырёх существ

Единственное правило программы трактуется так:
Если произвольно взятое существо X пернато и ещё не вымерло, то оно обязательно птица

Запрос к программе трактуется так:
Для какого существа X верно, что оно птица и летает?

ХЛП: декларативная семантика

Пример

Программа $\mathcal{P}$:

пернатый(**орёл**); есть_живой(**орёл**); летает(**орёл**);
пернатый(**чайка**); есть_живой(**чайка**); летает(**чайка**);
пернатый(**пингвин**); есть_живой(**пингвин**);
пернатый(**велоцираптор**);
птица(X) $\leftarrow$ пернатый(X), есть_живой(X);

Запрос $\mathcal{Q}$:

$$?\text{птица}(\texttt{X}), \text{летает}(\texttt{X})$$

Существует ровно два правильных ответа на запрос $\mathcal{Q}$ к программе $\mathcal{P}$:

$\{\texttt{X}/$**орёл**$\}$ и $\{\texttt{X}/$**чайка**$\}$

То есть верно

$$S_{\mathcal{P}} \models (\text{птица}(\texttt{X}) \,\&\, \text{летает}(\texttt{X}))\{\texttt{X}/\textbf{орёл}\} \text{ и}$$
$$S_{\mathcal{P}} \models (\text{птица}(\texttt{X}) \,\&\, \text{летает}(\texttt{X}))\{\texttt{X}/\textbf{чайка}\},$$

а для других ответов на $\mathcal{Q}$ аналогичные соотношения неверны

ХЛП: декларативная семантика

Пример

Программа $\mathcal{P}$:

пернатый(**орёл**); пернатый(**чайка**); пернатый(**пингвин**); пернатый(**велоцираптор**);

есть_живой(**орёл**); есть_живой(**чайка**); есть_живой(**пингвин**);

летает(**орёл**); летает(**чайка**);

птица(X) $\leftarrow$ пернатый(X), есть_живой(X);

Запрос $\mathcal{Q}$:

$$?\text{птица}(\text{X}), \text{летает}(\text{X})$$

А на запрос «?птица(**орёл**), летает(**орёл**)» к $\mathcal{P}$ есть только один правильный ответ: ε (тождественная подстановка)
Потому что согласно $\mathcal{P}$, орёл — это действительно летающая птица

А на запрос «?птица(**пингвин**), летает(**пингвин**)» к $\mathcal{P}$ нет ни одного правильного ответа
Потому что из $\mathcal{P}$ невозможно достоверно заключить, что пингвин летает

ХЛП: декларативная семантика

Другой пример: снова Даша, Саша, Паша и пиво

Программа $\mathcal{P}$:

$$\text{любит}(\textbf{даша}, \textbf{саша}); \quad \text{любит}(\textbf{саша}, \textbf{пиво}); \quad \text{любит}(\textbf{паша}, \textbf{пиво});$$
$$\text{любит}(\textbf{паша}, \mathtt{X}) \leftarrow \text{любит}(\textbf{паша}, \mathtt{Y}), \text{любит}(\mathtt{X}, \mathtt{Y});$$

Запрос $\mathcal{Q}$:

$$?\text{любит}(\mathtt{X}, \textbf{даша})$$

Единственное правило программы трактуется так:

Для любых произвольно взятых предметов X, Y, если и Паша, и X любят Y, то Паша любит X

Запрос к программе трактуется так:

Какие предметы X обязательно любят Дашу?

Существует ровно один правильный ответ на запрос $\mathcal{Q}$ к программе $\mathcal{P}$:

$$\{\mathtt{X}/\textbf{паша}\}$$

То есть $\{\mathtt{X}/\textbf{паша}\}$ — единственный ответ θ на $\mathcal{Q}$, для которого верно

$$S_{\mathcal{P}} \models \forall\ldots\ (\text{любит}(\mathtt{X}, \textbf{даша})\theta)$$

Математическая логика и логическое программирование

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

Блок 32

Хорновские логические программы: списки

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Список — это популярная структура данных, представляющая конечные последовательности элементов тех или иных множеств

В некоторых языках используется широкое понимание списка, согласно которому элементом может быть всё, что можно записать

В частности, элементами списков могут быть и другие списки

Чтобы избежать аналога парадокса Рассела для списков, спискам принято придавать особую характеристику — глубину (вложенности одних списков в другие), и разрешать в качестве элементов списка использовать только списки меньшей глубины

Здесь обойдёмся без понятия глубины за счёт неявного использования глубины в БНФ

Определение списка

Список L *сигнатуры* σ *логики предикатов* задаётся следующей БНФ:

$$L \ ::= \ \mathbf{nil} \ \mid \ t.L,$$

где $t \in \mathrm{Term}$

nil — это пустой список: особая константа, представляющая пустую последовательность

Головой непустой последовательности $\mathcal{S}$ будем называть первый элемент, а хвостом — последовательность, получающуюся из $\mathcal{S}$ удалением головы

. — это конструктор списков: особый двуместный функциональный символ, позволяющий описывать списки большего размера на основе списков меньшего размера в инфиксной форме

Символ **.** считается ассоциативным вправо: $a.b.c = a.(b.c)$

Списком $t.L$ представлена последовательность с головой t и хвостом L

Терминологию, применяющуюся к последовательностям, будем применять и к спискам, представляющим эти последовательности

Например, в записи «$t.L$» t — это голова и L — это хвост списка

Примеры списков

Последовательность элементов $a_1, \ldots, a_k$ будем в примерах обрамлять в скобки: $(a_1, \ldots, a_k)$

Пустую последовательность будем изображать как «пустую» пару скобок: ()

Примеры списков

Список	Соответствующая последовательность
nil	()
a.nil	(**a**)
a.X.b.nil	(**a**, X, **b**)
a.X.b	Это корректный терм, но не список
nil.nil	(())
nil.nil.nil	((), ())
(**nil.nil**).**nil**	((()))
(**a.b.nil**).(**c.d.nil**).**nil**	((**a**, **b**), (**c**, **d**))
((**a.b.nil**).**nil.nil**).(**nil.nil**).**nil**	(((**a**, **b**), ()), (()))

Примеры списков

Если захотите транслировать списки из предложенных обозначений в язык Prolog, то достаточно сделать следующее:

1. Вместо «**nil**» писать «[]»
2. Вместо «$a_1.\cdots.a_k.L$» писать «$[a_1, \ldots, a_k \mid L]$», соответственно транслировав $a_1, \ldots, a_k$ и L
3. По желанию вместо «$[a_1, \ldots, a_k \mid []]$» писать просто «$[a_1, \ldots, a_k]$»

Например:

Список в ХЛП	Список в Prolog
nil	[]
a.X**.b.nil**	[**a**, X, **b**] или [**a**, X, **b** | []]
nil.nil.nil	[[], []]
(**nil.nil**)**.nil**	[[[]]]
(**a.b.nil**)**.**(**c.d.nil**)**.nil**	[[**a**, **b**], [**c**, **d**]]
((**a.b.nil**)**.nil.nil**)**.**(**nil.nil**)**.nil**	[[[**a**, **b**], []], [[]]]
((**a.b.nil**)**.nil.nil**)**.**(**nil.nil**)**.**ℓ	[[[**a**, **b**], []], [[]] | ℓ]

Программы со списками

Пример

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

В этой программе записано *индуктивное определение* отношения $\mathrm{elem}(x, \ell)$ «x является элементом списка ℓ»:

1. Голова списка является его элементом
2. Если $\mathtt{X}$ является элементом хвоста списка, то $\mathtt{X}$ является элементом списка

Несложно видеть, что этими двумя пунктами «покрываются» все случаи принадлежности элемента списку

Правильными ответами на запрос
?$\mathrm{elem}(\mathtt{X}, \mathbf{тропинка.лесок.колосок.колосок.речка.небо\_голубое.nil})$
являются все подстановки элементов списка на место $\mathtt{X}$:
$\{\mathtt{X}/\mathbf{тропинка}\}$ $\{\mathtt{X}/\mathbf{лесок}\}$ $\{\mathtt{X}/\mathbf{колосок}\}$ $\{\mathtt{X}/\mathbf{речка}\}$ $\{\mathtt{X}/\mathbf{небо\_голубое}\}$

Программы со списками

Декларативная семантика подпрограмм естественным образом используется в декларативной семантике «над»программ

Например:

$$\mathrm{elem}(\mathtt{X}, \mathtt{X.L});$$
$$\mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$\mathrm{common}(\mathtt{X}, \mathtt{L}_1, \mathtt{L}_2) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L}_1), \mathrm{elem}(\mathtt{X}, \mathtt{L}_2);$$

Согласно первым двум правилам, запись $\mathrm{elem}(x, \ell)$ означает, что x является элементом списка ℓ

Значит $\mathrm{common}(\mathtt{X}, \mathtt{L}_1, \mathtt{L}_2)$ означает, что
$\mathtt{X}$ — элемент списка $\mathtt{L}_1$ и $\mathtt{X}$ — элемент списка $\mathtt{L}_2$

То есть
$\mathtt{X}$ — общий элемент списков $\mathtt{L}_1$ и $\mathtt{L}_2$

В частности, вот все правильные ответы на запрос

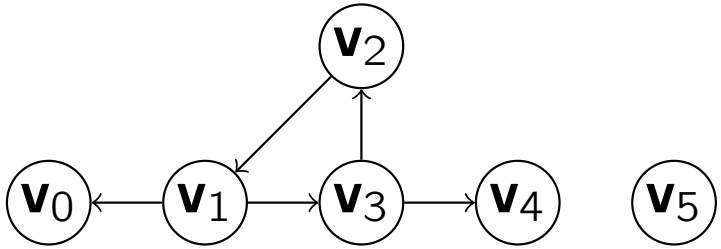
$$?\mathrm{common}(\mathtt{X}, \textbf{п.о.п.nil}, \textbf{к.л.о.п.nil})$$

к этой программе:

$$\{\mathtt{X}/\textbf{п}\} \qquad\qquad \{\mathtt{X}/\textbf{о}\}$$

Программы со списками

И ещё пример: задача о проверке достижимости в графе из блока 30



$\mathbf{v}_1 \rightsquigarrow \mathbf{v}_4$?

Путь $\mathbf{w}_1 \hookrightarrow \ldots \hookrightarrow \mathbf{w}_k$ можно представить как последовательность $(\mathbf{w}_1, \ldots, \mathbf{w}_k)$, и её — как список $\mathbf{w}_1\mathbf{.}\cdots\mathbf{.w}_k\mathbf{.nil}$

Систему дизъюнктов из блока 30 можно переписать как такие логическую программу и запрос:

$$\mathrm{arc}(\mathbf{v}_1, \mathbf{v}_0); \quad \mathrm{arc}(\mathbf{v}_1, \mathbf{v}_3); \quad \mathrm{arc}(\mathbf{v}_3, \mathbf{v}_2);$$
$$\mathrm{arc}(\mathbf{v}_2, \mathbf{v}_1); \quad \mathrm{arc}(\mathbf{v}_3, \mathbf{v}_4); \quad \mathrm{reach}(\mathtt{X}, \mathtt{X}, \mathtt{X}\mathbf{.nil});$$
$$\mathrm{reach}(\mathtt{X}, \mathtt{Z}, \mathtt{X.P}) \leftarrow \mathrm{arc}(\mathtt{X}, \mathtt{Y}), \mathrm{reach}(\mathtt{Y}, \mathtt{Z}, \mathtt{P}); \qquad ?\mathrm{reach}(\mathbf{v}_1, \mathbf{v}_4, \mathtt{P})$$

Существует бесконечно много правильных ответов на этот запрос к этой программе, отвечающих всевозможным путям из $\mathbf{v}_1$ в $\mathbf{v}_4$, — например:

$$\{\mathtt{P}/\mathbf{v}_1\mathbf{.}\mathbf{v}_3\mathbf{.}\mathbf{v}_4\mathbf{.nil}\}$$
$$\{\mathtt{P}/\mathbf{v}_1\mathbf{.}\mathbf{v}_3\mathbf{.}\mathbf{v}_2\mathbf{.}\mathbf{v}_1\mathbf{.}\mathbf{v}_3\mathbf{.}\mathbf{v}_4\mathbf{.nil}\}$$
$$\{\mathtt{P}/\mathbf{v}_1\mathbf{.}\mathbf{v}_3\mathbf{.}\mathbf{v}_2\mathbf{.}\mathbf{v}_1\mathbf{.}\mathbf{v}_3\mathbf{.}\mathbf{v}_2\mathbf{.}\mathbf{v}_1\mathbf{.}\mathbf{v}_3\mathbf{.}\mathbf{v}_4\mathbf{.nil}\}$$

Математическая логика и логическое программирование

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

Блок 33

Хорновские логические программы:
операционная семантика,
SLD-вычислимые ответы

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Программа $\mathcal{P}$:

$$\mathrm{elem}(\mathtt{X}, \mathtt{X.L});$$
$$\mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$\mathrm{common}(\mathtt{X}, \mathtt{L}_1, \mathtt{L}_2) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L}_1), \mathrm{elem}(\mathtt{X}, \mathtt{L}_2);$$

Запрос $\mathcal{Q}$:

$$?\mathrm{common}(\mathtt{X}, \textbf{п.о.п.nil}, \textbf{к.л.о.п.nil})$$

Правильными ответами на запрос $\mathcal{Q}$ к программе $\mathcal{P}$ являются подстановки

$$\{\mathtt{X}/\textbf{п}\} \qquad\qquad \{\mathtt{X}/\textbf{о}\}$$

А как можно **получить** (**извлечь**, **вычислить**) эти ответы?

ХЛП и хорновские дизъюнкты

$$\mathrm{elem}(\mathtt{X}, \mathtt{X.L}); \quad \mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$\mathrm{common}(\mathtt{X}, \mathtt{L}_1, \mathtt{L}_2) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L}_1), \mathrm{elem}(\mathtt{X}, \mathtt{L}_2);$$
$$?\mathrm{common}(\mathtt{X}, \textbf{п.о.п.nil}, \textbf{к.л.о.п.nil})$$

Согласно декларативной семантике, программа $\mathcal{P}$ представляет собой систему дизъюнктов-правил

$$S_{\mathcal{P}} = \left\{ \begin{array}{l} \forall \mathtt{X}\ \forall \mathtt{L}\ \mathrm{elem}(\mathtt{X}, \mathtt{X.L}), \quad \forall \mathtt{X}\ \forall \mathtt{L}\ \forall \mathtt{Y}\ (\mathrm{elem}(\mathtt{X}, \mathtt{L}) \to \mathrm{elem}(\mathtt{X}, \mathtt{Y.L})), \\ \forall \mathtt{X}\ \forall \mathtt{L}_1\ \forall \mathtt{L}_2\ (\mathrm{elem}(\mathtt{X}, \mathtt{L}_1)\ \&\ \mathrm{elem}(\mathtt{X}, \mathtt{L}_2) \to \mathrm{common}(\mathtt{X}, \mathtt{L}_1, \mathtt{L}_2)) \end{array} \right\},$$

запрос $\mathcal{Q}$ представляет собой формулу

$$\Phi_{\mathcal{Q}} = \mathrm{common}(\mathtt{X}, \textbf{п.о.п.nil}, \textbf{к.л.о.п.nil}),$$

и правильность ответа θ означает, что справедливо соотношение

$$S_{\mathcal{P}} \models \forall \ldots\ (\Phi_{\mathcal{Q}}\theta)$$

Формула $\Phi_{\mathcal{Q}}$ — это отрицание дизъюнкта-запроса:

$$D_{\mathcal{Q}} = \neg\mathrm{common}(\mathtt{X}, \textbf{п.о.п.nil}, \textbf{к.л.о.п.nil}) = \neg\Phi_{\mathcal{Q}}$$

$$S_{\mathcal{P}} \models \forall \ldots\ (\Phi_{\mathcal{Q}}\theta) \quad \Leftrightarrow \quad S_{\mathcal{P}} \models \forall \ldots\ (\neg D_{\mathcal{Q}}\theta)$$

ХЛП и хорновские дизъюнкты

$$\mathrm{elem}(\mathtt{X}, \mathtt{X.L}); \quad \mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$\mathrm{common}(\mathtt{X}, \mathrm{L}_1, \mathrm{L}_2) \leftarrow \mathrm{elem}(\mathtt{X}, \mathrm{L}_1), \mathrm{elem}(\mathtt{X}, \mathrm{L}_2);$$
$$?\mathrm{common}(\mathtt{X}, \textbf{п.о.п.nil}, \textbf{к.л.о.п.nil})$$

$$S_{\mathcal{P}} = \left\{ \begin{array}{l} \mathrm{elem}(\mathtt{X}, \mathtt{X.L}), \quad \mathrm{elem}(\mathtt{X}, \mathtt{L}) \to \mathrm{elem}(\mathtt{X}, \mathtt{Y.L}), \\ \mathrm{elem}(\mathtt{X}, \mathrm{L}_1) \,\&\, \mathrm{elem}(\mathtt{X}, \mathrm{L}_2) \to \mathrm{common}(\mathtt{X}, \mathrm{L}_1, \mathrm{L}_2)) \end{array} \right\}$$

$$S_{\mathcal{P}} \models \forall \ldots \, (\Phi_{\mathcal{Q}}\theta)$$

Способ извлечения требуемой подстановки θ для получившихся системы дизъюнктов-правил и дизъюнкта-запроса был сформулирован в теореме о входном резолютивном выводе: достаточно построить входной вывод из $S_{\mathcal{P}} \cup \{D_{\mathcal{Q}}\}$, инициированный дизъюнктом $D_{\mathcal{Q}}$, и вычислить композицию подстановок этого вывода

Перепишем этот способ в терминах логических программ

ХЛП и хорновские дизъюнкты

Результаты $\mathcal{R}\theta$, $\mathcal{Q}\theta$ применения подстановки θ соответственно к правилу $\mathcal{R} = A \leftarrow B_1, \ldots, B_k$ и к запросу $\mathcal{Q} = ?C_1, \ldots, C_m$ определим так:

$$\mathcal{R}\theta = A\theta \leftarrow B_1\theta, \ldots, B_k\theta;$$
$$\mathcal{Q}\theta = ?C_1\theta, \ldots, C_m\theta$$

$\mathrm{Var}_{\mathcal{R}}$ и $\mathrm{Var}_{\mathcal{Q}}$ — так обозначим все переменные, содержащиеся в правиле $\mathcal{R}$ и в запросе $\mathcal{Q}$

Терминология, относящаяся к применению подстановок к выражениям (вариант, пример, основной пример, основное правило, основной запрос), без изменений переносится с дизъюнктов на правила и запросы

Утверждение

Для любых правил $\mathcal{R}$, $\mathcal{R}'$ и любой подстановки θ верно:

$$\mathcal{R}' = \mathcal{R}\theta \quad \Leftrightarrow \quad \Phi_{\mathcal{R}'} = \Phi_{\mathcal{R}}\theta$$

Утверждение

Для любых запросов $\mathcal{Q}$, $\mathcal{Q}'$ и любой подстановки θ верно:

$$\mathcal{Q}' = \mathcal{Q}\theta \quad \Leftrightarrow \quad \Phi_{\mathcal{Q}'} = \Phi_{\mathcal{Q}}\theta \quad \Leftrightarrow \quad \Phi_{\neg\mathcal{Q}'} = \Phi_{\neg\mathcal{Q}}\theta$$

Правило SLD-резолюции

Пусть

- $\mathcal{Q} = ?C_1, \ldots, C_{i-1}, C_i, C_{i+1}, \ldots, C_m$ — запрос
- $\mathcal{R} = A \leftarrow B_1, \ldots, B_k;$ — правило
- $\mathcal{R}' = \mathcal{R}\eta$ — вариант правила $\mathcal{R}$ для переименования η, не содержащий ни одной переменной из $\mathcal{Q}$
- $\theta \in \text{НОУ}(C_i, A\eta)$

Тогда запрос

$$\mathcal{Q}' = (?C_1, \ldots, C_{i-1}, B_1\eta, \ldots B_k\eta, C_{i+1}, \ldots, C_m)\theta$$

называется SLD-резольвентой запроса $\mathcal{Q}$ и правила $\mathcal{R}$ для подцели C_i и унификатора θ

То, что $\mathcal{Q}'$ — SLD-резольвента $\mathcal{Q}$ и $\mathcal{R}$ как написано выше, будем обозначать записями

$$\mathcal{Q} \xrightarrow{\mathcal{R}\eta, i, \theta} \mathcal{Q}' \quad \text{и} \quad \begin{array}{ccc} \mathcal{Q} & \xrightarrow{i,\,\theta} & \\ & \searrow & \mathcal{Q}' \\ \mathcal{R}\eta & \nearrow & \end{array}$$

Правило SLD-резолюции

Примеры

$$?\mathrm{p}(\mathtt{X}), \mathrm{r}(\mathtt{X}, \mathbf{f}(\mathtt{Y})), \mathrm{r}(\mathtt{Y}, \mathbf{c})$$

$$2 \left| \begin{array}{l} \mathrm{r}(\mathbf{d}, \mathtt{X}') \leftarrow \mathrm{p}(\mathtt{X}'), \mathrm{r}(\mathbf{c}, \mathtt{Y}'); \\ \{\mathtt{X}/\mathbf{d}, \mathtt{X}'/\mathbf{f}(\mathtt{Y})\} \end{array} \right.$$

$$?\mathrm{p}(\mathbf{d}), \mathrm{p}(\mathbf{f}(\mathtt{Y})), \mathrm{r}(\mathbf{c}, \mathtt{Y}'), \mathrm{r}(\mathtt{Y}, \mathbf{c})$$

$$?\mathrm{p}(\mathtt{X}), \mathrm{r}(\mathtt{X}, \mathbf{f}(\mathtt{Y})), \mathrm{r}(\mathtt{Y}, \mathbf{c})$$

$$3 \left| \begin{array}{l} \mathrm{r}(\mathbf{d}, \mathtt{X}') \leftarrow \mathrm{p}(\mathtt{X}'), \mathrm{r}(\mathbf{c}, \mathtt{Y}'); \\ \{\mathtt{Y}/\mathbf{d}, \mathtt{X}'/\mathbf{c}\} \end{array} \right.$$

$$?\mathrm{p}(\mathtt{X}), \mathrm{r}(\mathtt{X}, \mathbf{f}(\mathbf{d})), \mathrm{p}(\mathbf{c}), \mathrm{r}(\mathbf{c}, \mathtt{Y}')$$

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

$$1 \left| \begin{array}{l} \mathrm{r}(\mathtt{X}'\mathbf{.nil}, \mathtt{Y}') \leftarrow; \\ \{\mathtt{X}/\mathtt{X}'\mathbf{.nil}, \mathtt{Y}'/(\mathtt{X}'\mathbf{.nil})\mathbf{.nil}\} \end{array} \right.$$

$$\square$$

А у запроса $?\mathrm{r}(\mathtt{X}, \mathtt{X}\mathbf{.nil})$ и правила $\mathrm{r}(\mathtt{X}\mathbf{.nil}, \mathtt{X}) \leftarrow$; нет SLD-резольвент

Утверждение. **Для любых запросов $\mathcal{Q}$ и $\mathcal{Q}'$ и правила $\mathcal{R}$ верно: $\mathcal{Q}'$ — SLD-резольвента $\mathcal{Q}$ и $\mathcal{R}$ $\Leftrightarrow$ $\neg\Phi_{\mathcal{Q}'}$ — резольвента $\neg\Phi_{\mathcal{Q}}$ и некоторого варианта дизъюнкта $\Phi_{\mathcal{R}}$ для того же выбора атомов и того же унификатора**

Правило SLD-резолюции

Немного о названии «SLD-резолюция»:

S = Selective L = Linear D = Definite

Но если Вы сейчас попытались представить, что может означать каждое из этих слов, и думаете, что преуспели, то скорее всего это верно лишь отчасти

SLD-резолюция была придумана шотландским математиком Робертом Ковальски в начале далёких 1970-х годов в качестве одной из первых попыток осмыслить логику предикатов как язык программирования

Буквы S, L и D — это ограничения на построение резолютивного вывода в терминах, использовавшихся в те далёкие годы

А сейчас достаточно просто запомнить эти три буквы, воспринимая их как историческое наследие

SLD-резолютивные вычисления

(Частичным) SLD-резолютивным вычислением программы $\mathcal{P}$, порождённым запросом $\mathcal{Q}$, называется последовательность запросов (конечная или бесконечная) вида

$$\mathcal{Q}_1 \xrightarrow{\mathcal{R}_1, k_1, \theta_1} \mathcal{Q}_2 \xrightarrow{\mathcal{R}_2, k_2, \theta_2} \dots ,$$

где:

- $\mathcal{Q}_1 = \mathcal{Q}$
- $\mathcal{R}_i$, k_i и θ_i, $i \in \{1, 2, \dots\}$ — соответственно вариант какого-либо правила из $\mathcal{P}$, номер подцели и унификатор для резольвенты $\mathcal{Q}_{i+1}$

По умолчанию в примерах будем использовать вариант $\mathcal{R}'$ правила $\mathcal{R}$, в котором ко всем переменным правила добавлены штрихи

Вычисление будем называть

- успешным, если оно конечно и оканчивается запросом $\square$
- тупиковым, если оно конечно, неуспешно и не может быть продолжено до более длинного вычисления

SLD-резолютивные вычисления

Для подстановки θ и множества переменных V записью $\theta|_V$ обозначим проекцию подстановки θ на множество V, то есть подстановку, устроенную так:

- Если $\mathrm{y} \in V$, то $\theta|_V(\mathrm{y}) = \theta(\mathrm{y})$
- Иначе $\theta|_V(\mathrm{y}) = \mathrm{y}$

Результатом конечного вычисления

$$\mathcal{Q}_1 \xrightarrow{\mathcal{R}_1, k_1, \theta_1} \ldots \xrightarrow{\mathcal{R}_{n-1}, k_{n-1}, \theta_{n-1}} \mathcal{Q}_n$$

,
а также ответом, SLD-вычисленным этим вычислением, будем называть подстановку $\theta_1 \ldots \theta_{n-1}|_{\mathrm{Var}_{\mathcal{Q}_1}}$

Ответ на запрос $\mathcal{Q}$ к программе $\mathcal{P}$ будем называть SLD-резолютивно вычислимым (или просто SLD-вычислимым), если существует успешное SLD-резолютивное вычисление, результатом которого является этот ответ

SLD-резолютивные вычисления

Примеры SLD-резолютивных вычислений программы $\mathcal{P}$

$\mathcal{R}_1$: $\mathrm{elem}(\mathtt{X}, \mathtt{X.L})$;

$\mathcal{R}_2$: $\mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;

$\mathcal{R}_3$: $\mathrm{common}(\mathtt{X}, \mathtt{L}_1, \mathtt{L}_2) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L}_1), \mathrm{elem}(\mathtt{X}, \mathtt{L}_2)$;

$?\mathrm{elem}(\mathtt{X}, \textbf{к.л.о.п.nil})$

$\mathcal{R}'_1, 1 \downarrow \theta_1 = \{\mathtt{X}'/\textbf{к}, \mathtt{L}'/\textbf{л.о.п.nil}, \mathtt{X}/\textbf{к}\}$

$\square$

Значит, подстановка $\theta_1|_{\{\mathtt{X}\}} = \{\mathtt{X}/\textbf{к}\}$ — SLD-вычислимый ответ на запрос $?\mathrm{elem}(\mathtt{X}, \textbf{к.л.о.п.nil})$ к $\mathcal{P}$

SLD-резолютивные вычисления

Примеры SLD-резолютивных вычислений программы $\mathcal{P}$

$$\mathcal{R}_1: \quad \mathrm{elem}(\mathtt{X}, \mathtt{X.L});$$
$$\mathcal{R}_2: \quad \mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$\mathcal{R}_3: \quad \mathrm{common}(\mathtt{X}, \mathtt{L}_1, \mathtt{L}_2) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L}_1), \mathrm{elem}(\mathtt{X}, \mathtt{L}_2);$$

$$?\mathrm{elem}(\mathtt{X}, \textbf{к.л.о.п.nil})$$
$$\mathcal{R}'_2, 1 \downarrow \theta_1 = \{\mathtt{X}'/\mathtt{X}, \mathtt{Y}'/\textbf{к}, \mathtt{L}'/\textbf{л.о.п.nil}\}$$
$$?\mathrm{elem}(\mathtt{X}, \textbf{л.о.п.nil})$$
$$\mathcal{R}'_2, 1 \downarrow \theta_2 = \{\mathtt{X}'/\mathtt{X}, \mathtt{Y}'/\textbf{л}, \mathtt{L}'/\textbf{о.п.nil}\}$$
$$?\mathrm{elem}(\mathtt{X}, \textbf{о.п.nil})$$
$$\mathcal{R}'_1, 1 \downarrow \theta_3 = \{\mathtt{X}'/\textbf{о}, \mathtt{L}'/\textbf{п.nil}, \mathtt{X}/\textbf{о}\}$$
$$\square$$

Значит, подстановка $\theta_1\theta_2\theta_3|_{\{\mathtt{X}\}} = \{\mathtt{X}/\textbf{о}\}$ — SLD-вычислимый ответ на запрос $?\mathrm{elem}(\mathtt{X}, \textbf{к.л.о.п.nil})$ к $\mathcal{P}$

SLD-резолютивные вычисления

Примеры SLD-резолютивных вычислений программы $\mathcal{P}$

$\mathcal{R}_1$: $\mathrm{elem}(\mathtt{X}, \mathtt{X.L})$;

$\mathcal{R}_2$: $\mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L})$;

$\mathcal{R}_3$: $\mathrm{common}(\mathtt{X}, \mathtt{L}_1, \mathtt{L}_2) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L}_1), \mathrm{elem}(\mathtt{X}, \mathtt{L}_2)$;

$?\mathrm{common}(\mathtt{X}, \textbf{п.о.п.nil}, \textbf{к.л.о.п.nil})$

$\mathcal{R}'_3, 1 \downarrow \theta_1 = \{\mathtt{X}'/\mathtt{X}, \mathtt{L}'_1/\textbf{п.о.п.nil}, \mathtt{L}'_2/\textbf{к.л.о.п.nil}\}$

$?\mathrm{elem}(\mathtt{X}, \textbf{п.о.п.nil}), \mathrm{elem}(\mathtt{X}, \textbf{к.л.о.п.nil})$

$\mathcal{R}'_2, 2 \downarrow \theta_2 = \{\mathtt{X}'/\mathtt{X}, \mathtt{Y}'/\textbf{к}, \mathtt{L}'/\textbf{л.о.п.nil}\}$

$?\mathrm{elem}(\mathtt{X}, \textbf{п.о.п.nil}), \mathrm{elem}(\mathtt{X}, \textbf{л.о.п.nil})$

$\mathcal{R}'_2, 1 \downarrow \theta_3 = \{\mathtt{X}'/\mathtt{X}, \mathtt{Y}'/\textbf{п}, \mathtt{L}'/\textbf{о.п.nil}\}$

$?\mathrm{elem}(\mathtt{X}, \textbf{о.п.nil}), \mathrm{elem}(\mathtt{X}, \textbf{л.о.п.nil})$

$\mathcal{R}'_1, 1 \downarrow \theta_4 = \{\mathtt{X}'/\textbf{о}, \mathtt{L}'/\textbf{п.nil}, \mathtt{X}/\textbf{о}\}$

$?\mathrm{elem}(\textbf{о}, \textbf{л.о.п.nil})$

$\mathcal{R}'_2, 1 \downarrow \theta_5 = \{\mathtt{X}'/\textbf{о}, \mathtt{Y}'/\textbf{л}, \mathtt{L}'/\textbf{о.п.nil}\}$

$?\mathrm{elem}(\textbf{о}, \textbf{о.п.nil})$

$\mathcal{R}'_1, 1 \downarrow \theta_6 = \{\mathtt{X}'/\textbf{о}, \mathtt{L}'/\textbf{п.nil}\}$

$\square$

Значит, подстановка $\theta_1\theta_2\theta_3\theta_4\theta_5\theta_6|_{\{\mathtt{X}\}} = \{\mathtt{X}/\textbf{о}\}$ — SLD-вычислимый ответ на запрос $?\mathrm{common}(\mathtt{X}, \textbf{п.о.п.nil}, \textbf{к.л.о.п.nil})$ к $\mathcal{P}$

SLD-резолютивные вычисления

Примеры SLD-резолютивных вычислений программы $\mathcal{P}$

$$\mathcal{R}_1: \quad \mathrm{elem}(\mathtt{X}, \mathtt{X.L});$$
$$\mathcal{R}_2: \quad \mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$$
$$\mathcal{R}_3: \quad \mathrm{common}(\mathtt{X}, \mathtt{L}_1, \mathtt{L}_2) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L}_1), \mathrm{elem}(\mathtt{X}, \mathtt{L}_2);$$

$$?\mathrm{common}(\mathtt{X}, \textbf{п.о.п.nil}, \textbf{к.л.о.п.nil})$$
$$\mathcal{R}_3', 1{\downarrow}\theta_1 = \{\mathtt{X}'/\mathtt{X}, \mathtt{L}_1'/\textbf{п.о.п.nil}, \mathtt{L}_2'/\textbf{к.л.о.п.nil}\}$$
$$?\mathrm{elem}(\mathtt{X}, \textbf{п.о.п.nil}), \mathrm{elem}(\mathtt{X}, \textbf{к.л.о.п.nil})$$
$$\mathcal{R}_1', 2{\downarrow}\theta_2 = \{\mathtt{X}'/\textbf{к}, \mathtt{L}'/\textbf{л.о.п.nil}, \mathtt{X}/\textbf{к}\}$$
$$?\mathrm{elem}(\textbf{к}, \textbf{п.о.п.nil})$$
$$\mathcal{R}_2', 1{\downarrow}\theta_3 = \{\mathtt{X}'/\textbf{к}, \mathtt{Y}'/\textbf{п}, \mathtt{L}'/\textbf{о.п.nil}\}$$
$$?\mathrm{elem}(\textbf{к}, \textbf{о.п.nil})$$
$$\mathcal{R}_2', 1{\downarrow}\theta_4 = \{\mathtt{X}'/\textbf{к}, \mathtt{Y}'/\textbf{о}, \mathtt{L}'/\textbf{п.nil}\}$$
$$?\mathrm{elem}(\textbf{к}, \textbf{п.nil})$$
$$\mathcal{R}_2', 1{\downarrow}\theta_5 = \{\mathtt{X}'/\textbf{к}, \mathtt{Y}'/\textbf{п}, \mathtt{L}'/\textbf{nil}\}$$
$$?\mathrm{elem}(\textbf{к}, \textbf{nil})$$

Это тупиковое вычисление, им не задаётся SLD-вычислимый ответ

Математическая логика и логическое программирование

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

Блок 34

Хорновские логические программы:
корректность операционной семантики

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Для ХЛП определены две семантики:

1. Декларативная (основная):
   - Программа — это система дизъюнктов-правил
   - Правильный ответ — это подстановка целевых переменных, при применении которой к запросу он следует из программы

2. Операционная (вспомогательная):
   - Шаг вычисления программы — это применение правила SLD-резолюции
   - SLD-вычислимый ответ — это композиция подстановок, получаемых по ходу вычисления, спроектированная на целевые переменные

А как связаны между собой правильные и SLD-вычислимые ответы?

Лемма о соответствии вычислений и входных выводов

Для любого конечного SLD-резолютивного вычисления

$$\mathcal{Q}_1 \xrightarrow{\mathcal{R}_1, k_1, \theta_1} \mathcal{Q}_2 \xrightarrow{\mathcal{R}_2, k_2, \theta_2} \ldots \xrightarrow{\mathcal{R}_{n-1}, k_{n-1}, \theta_{n-1}} \mathcal{Q}_n$$

программы $\mathcal{P}$ последовательность дизъюнктов

$$\neg\Phi_{\mathcal{Q}_1}, \Phi_{\mathcal{R}_1}, \neg\Phi_{\mathcal{Q}_2}, \Phi_{\mathcal{R}_2}, \ldots, \Phi_{\mathcal{R}_{n-1}}, \neg\Phi_{\mathcal{Q}_n}$$

является входным резолютивным выводом из $S_{\mathcal{P}} \cup \{\neg\Phi_{\mathcal{Q}_1}\}$, в котором для вычисления резольвент $\mathcal{Q}_2, \ldots, \mathcal{Q}_n$ применяются соответственно унификаторы $\theta_1, \ldots, \theta_{n-1}$

Доказательство.

База: $n = 1$

$\neg\Phi_{\mathcal{Q}_1}$ — это тривиальный входной вывод из $S_{\mathcal{P}} \cup \{\neg\Phi_{\mathcal{Q}_1}\}$

Индуктивный переход: полагая утверждение доказанным для $n < N$, докажем его для $n = N$

Лемма о соответствии вычислений и входных выводов

Для любого конечного SLD-резолютивного вычисления

$$\mathcal{Q}_1 \xrightarrow{\mathcal{R}_1, k_1, \theta_1} \mathcal{Q}_2 \xrightarrow{\mathcal{R}_2, k_2, \theta_2} \ldots \xrightarrow{\mathcal{R}_{n-1}, k_{n-1}, \theta_{n-1}} \mathcal{Q}_n$$

программы $\mathcal{P}$ последовательность дизъюнктов

$$\neg\Phi_{\mathcal{Q}_1}, \Phi_{\mathcal{R}_1}, \neg\Phi_{\mathcal{Q}_2}, \Phi_{\mathcal{R}_2}, \ldots, \Phi_{\mathcal{R}_{n-1}}, \neg\Phi_{\mathcal{Q}_n}$$

является входным резолютивным выводом из $S_{\mathcal{P}} \cup \{\neg\Phi_{\mathcal{Q}_1}\}$, в котором для вычисления резольвент $\mathcal{Q}_2, \ldots, \mathcal{Q}_n$ применяются соответственно унификаторы $\theta_1, \ldots, \theta_{n-1}$

Доказательство (индуктивный переход).

По индуктивному предположению, последовательность

$$\neg\Phi_{\mathcal{Q}_2}, \Phi_{\mathcal{R}_2}, \ldots, \Phi_{\mathcal{R}_{n-1}}, \neg\Phi_{\mathcal{Q}_n}$$

является входным резолютивным выводом из $S_{\mathcal{P}} \cup \{\neg\Phi_{\mathcal{Q}_2}\}$, в котором последовательно применяются унификаторы $\theta_2 \ldots \theta_{n-1}$

Как обсуждалось ранее, $\neg\Phi_{\mathcal{Q}_2}$ — это резольвента дизъюнктов $\neg\Phi_{\mathcal{Q}_1}$ и $\Phi_{\mathcal{R}_1}$ для унификатора θ_1

Осталось заметить, что $\Phi_{\mathcal{R}_1}$ — вариант дизъюнкта из $S_{\mathcal{P}}$ ▼

Теорема о корректности операционной семантики ХЛП

Для любой ХЛП $\mathcal{P}$ и любого запроса $\mathcal{Q}$ верно следующее: любой SLD-вычислимый ответ на $\mathcal{Q}$ к $\mathcal{P}$ является правильным ответом на $\mathcal{Q}$ к $\mathcal{P}$

Доказательство.

Рассмотрим SLD-вычислимый ответ θ на запрос $\mathcal{Q}$ к $\mathcal{P}$

По определению этого ответа, существует успешное SLD-резолютивное вычисление

$$\mathcal{Q}_1 \xrightarrow{\mathcal{R}_1, k_1, \theta_1} \mathcal{Q}_2 \xrightarrow{\mathcal{R}_2, k_2, \theta_2} \ldots \xrightarrow{\mathcal{R}_n, k_n, \theta_n} \square,$$

такое что $\mathcal{Q}_1 = \mathcal{Q}$ и $\theta = (\theta_1\theta_2\ldots\theta_n)|_{\mathrm{Var}_{\mathcal{Q}}}$

По последней лемме, последовательность

$$\neg\Phi_{\mathcal{Q}_1}, \Phi_{\mathcal{R}_1}, \neg\Phi_{\mathcal{Q}_2}, \Phi_{\mathcal{R}_2}, \ldots, \Phi_{\mathcal{R}_n}, \square$$

является входным резолютивным выводом из $S_{\mathcal{P}} \cup \{\neg\Phi_{\mathcal{Q}_1}\}$, в котором для построения резольвент применяются унификаторы $\theta_1, \theta_2, \ldots, \theta_n$

Теорема о корректности операционной семантики ХЛП

Для любой ХЛП $\mathcal{P}$ и любого запроса $\mathcal{Q}$ верно следующее: любой SLD-вычислимый ответ на $\mathcal{Q}$ к $\mathcal{P}$ является правильным ответом на $\mathcal{Q}$ к $\mathcal{P}$

Доказательство. (SLD-вычислимый ответ $\theta = (\theta_1\theta_2\ldots\theta_n)|_{\mathrm{Var}_\mathcal{Q}}$)

По теореме о входном резолютивном выводе как средстве вычисления, верно соотношение

$$S_\mathcal{P} \models \forall\ldots\, (\Phi_\mathcal{Q}\theta_1\theta_2\ldots\theta_n)$$

По свойствам применения подстановок и устройству переменных запроса, справедливо и

$$S_\mathcal{P} \models \forall\ldots\, (\Phi_\mathcal{Q}(\theta_1\theta_2\ldots\theta_n)|_{\mathrm{Var}_\mathcal{Q}})$$

Значит, $(\theta_1\theta_2\ldots\theta_n)|_{\mathrm{Var}_\mathcal{Q}}$ — это по определению правильный ответ на запрос $\mathcal{Q}$ к программе $\mathcal{P}$ ▼

Математическая логика и логическое программирование

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

Блок 35

Хорновские логические программы:
полнота операционной семантики

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Для ХЛП определены две семантики:

1. Декларативная (основная):
   - Программа — это система формул
   - Правильный ответ — это подстановка целевых переменных, при применении которой к запросу он следует из программы

2. Операционная (вспомогательная):
   - Шаг вычисления программы — это применение правила SLD-резолюции
   - SLD-вычислимый ответ — это композиция подстановок, получаемых по ходу вычисления, спроектированная на целевые переменные

Показано, что каждый SLD-вычислимый ответ является правильным

А верно ли утверждение в обратную сторону?
(*О том, что любой правильный ответ обязательно является SLD-вычислимым*)

Особенности полноты операционной семантики

Рассмотрим такой **пример**:

$$\mathrm{p}(\mathbf{f}(\mathtt{X}), \mathbf{c}) \leftarrow \mathrm{q}(\mathtt{X});$$
$$\mathrm{q}(\mathtt{X});$$
$$?\mathrm{p}(\mathtt{X}, \mathtt{Y})$$

Правильным ответом на этот запрос к этой программе является, например, подстановка $\theta = \{\mathtt{X}/\mathbf{f}(\mathbf{c}), \mathtt{Y}/\mathbf{c}\}$

При этом единственный (*с точностью до переименования*) SLD-вычислимый ответ $\{\mathtt{X}/\mathbf{f}(\mathtt{X}), \mathtt{Y}/\mathbf{c}\}$ не совпадает с θ

Значит, *ожидаемое* утверждение «любой правильный ответ является SLD-вычислимым» заведомо неверно

Но упомянутый правильный ответ является *частным случаем* вычисленного:

$$\{\mathtt{X}/\mathbf{f}(\mathbf{c}), \mathtt{Y}/\mathbf{c}\} = \{\mathtt{X}/\mathbf{f}(\mathtt{X}), \mathtt{Y}/\mathbf{c}\}\{\mathtt{X}/\mathbf{c}\}$$

Докажем полноту операционной семантики с поправкой на это наблюдение

Общая схема обоснования полноты

Подстановку θ будем называть частным случаем подстановки η, а подстановку η — обобщением подстановки θ, если существует подстановка μ, такая что $\theta = \eta\mu$

Записью $[\mathcal{P}]$ обозначим *бесконечную* ХЛП, состоящую из всех основных примеров всех правил ХЛП $\mathcal{P}$ в произвольном порядке

Полнота операционной семантики будет обосновываться так:

1. Рассмотрим произвольный правильный ответ θ на запрос $\mathcal{Q}$ к программе $\mathcal{P}$
2. Покажем, как построить успешное вычисление ответа ε на особый основной запрос $\mathcal{Q}\theta\mu$ к бесконечной программе $[\mathcal{P}]$
   (лемма об основных вычислениях)
3. Покажем, как преобразовать построенное вычисление на запрос $\mathcal{Q}\theta\mu$ к $[\mathcal{P}]$ в успешное вычисление некоторого обобщения η ответа $\theta\mu$ на запрос $\mathcal{Q}$ к $\mathcal{P}$
   (лемма о подъёме вычисления)
4. Покажем, что ответ θ — частный случай ответа η

Лемма об основных вычислениях

Для любой ХЛП $\mathcal{P}$ и любого основного запроса $\mathcal{Q}$, такого что $S_{\mathcal{P}} \models \Phi_{\mathcal{Q}}$, существует успешое SLD-резолютивное вычисление, порождённое запросом $\mathcal{Q}$ к $[\mathcal{P}]$

Доказательство

Пусть $\mathcal{P} = (\mathcal{R}_1, \ldots, \mathcal{R}_n)$

Согласно теореме о логическом следствии, если $S_{\mathcal{P}} \models \Phi_{\mathcal{Q}}$, то формула $\Phi_{\mathcal{R}_1} \,\&\, \ldots \,\&\, \Phi_{\mathcal{R}_n} \to \Phi_{\mathcal{Q}}$ общезначима

Следовательно, система дизъюнктов $\{\Phi_{\mathcal{R}_1}, \ldots, \Phi_{\mathcal{R}_n}, \neg\Phi_{\mathcal{Q}}\}$, отвечающая отрицанию этой формулы, невыполнима

По теореме Эрбрана, существует конечный набор основных примеров $D_1, \ldots, D_m$ дизъюнктов из $\{D_{\mathcal{R}_1}, \ldots, D_{\mathcal{R}_n}\}$, такой что система основных дизъюнктов $S = \{D_1, \ldots, D_m, \neg\Phi_{\mathcal{Q}}\}$ невыполнима

Согласно соответствиям между запросами, правилами и дизъюнктами и между SLD-резолютивными вычислениями и резолютивными выводами, осталось показать, что существует входной вывод $\square$ из S, инициированный дизъюнктом $\neg\Phi_{\mathcal{Q}}$

Лемма об основных вычислениях (доказательство)

$S = \{D_1, \ldots, D_m, \neg\Phi_{\mathcal{Q}}\}$; $\quad S : \neg\Phi_{\mathcal{Q}} \rightsquigarrow \square$?

По теореме о полноте резолютивного вывода, существует вывод $\mathfrak{D}$ пустого дизъюнкта $\square$ из S

Но этот этот вывод не обязан быть входным, и тем более инициированным $\neg\Phi_{\mathcal{Q}}$

Осталось показать, как перестроить вывод $\mathfrak{D}$ во входной, инициированный $\neg\Phi_{\mathcal{Q}}$

В качестве отправной точки используем тот факт, что $\square$ может быть получен **только** как резольвента дизъюнкта-запроса и дизъюнкта-правила (но не обязательно из S), и значит, в частности, в выводе есть хотя бы один запрос

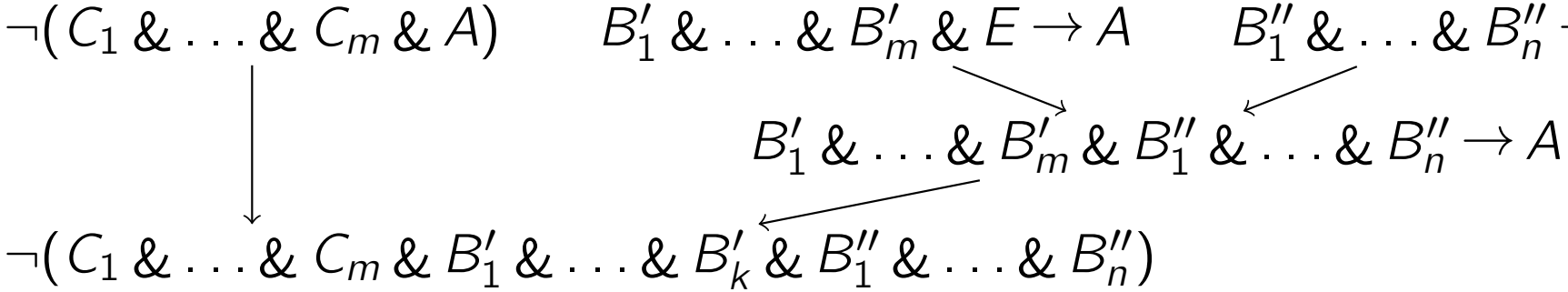
Проблема 1, которую требуется решить, чтобы вывод стал входным: в выводе могут строиться «лишние» резольвенты и склейки, т.е. такие, которые никак не связаны с получением $\square$

Очистим вывод от таких резольвент, склеек и вариантов, не использующихся для получения $\square$

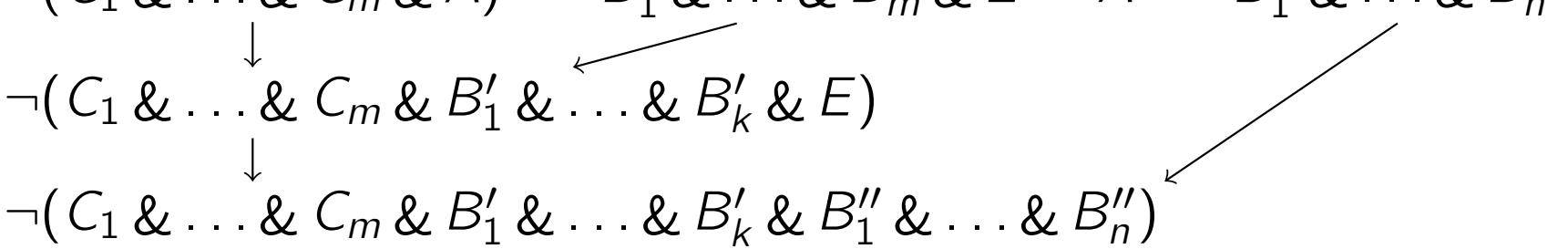
Лемма об основных вычислениях (доказательство)

$S = \{D_1, \ldots, D_m, \neg\Phi_{\mathcal{Q}}\}$; $\quad S : \neg\Phi_{\mathcal{Q}} \rightsquigarrow \square$?

Проблема 2: в выводе может строиться резольвента дизъюнкта-запроса и резольвенты двух дизъюнктов-правил:



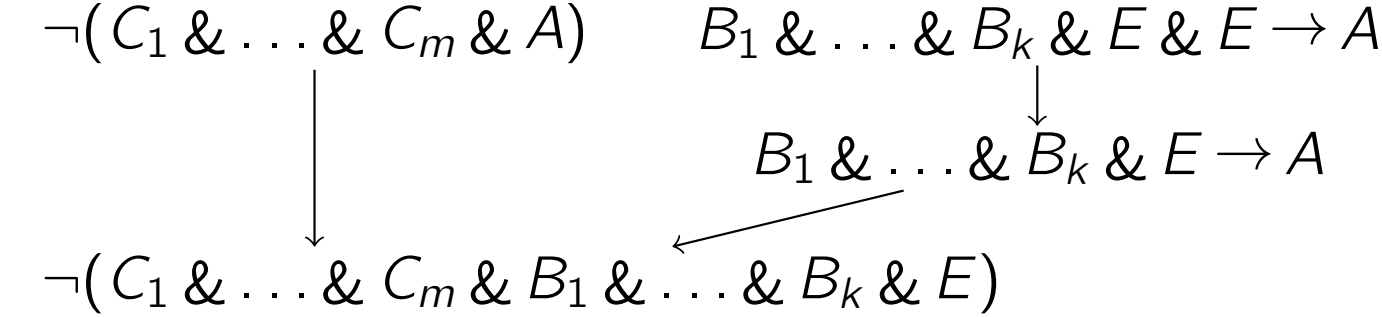
Эту проблему можно решить, перестроив вывод:



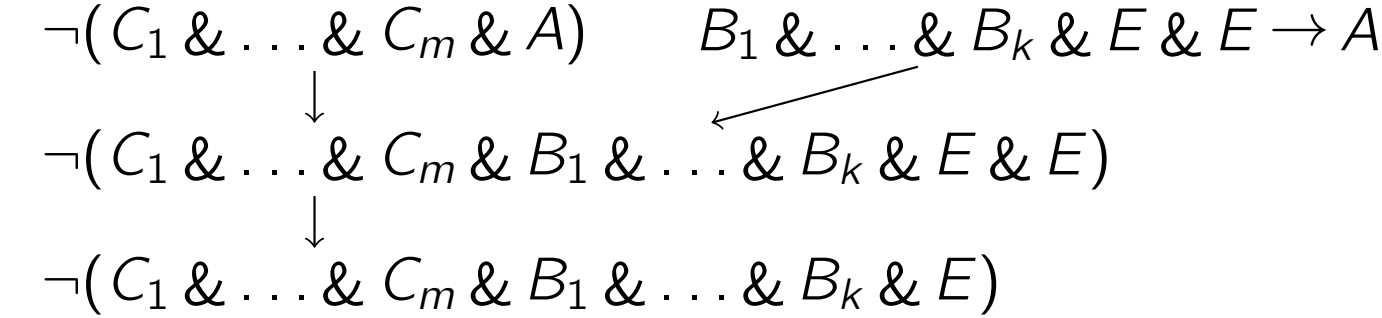
Лемма об основных вычислениях (доказательство)

$S = \{D_1, \ldots, D_m, \neg\Phi_{\mathcal{Q}}\}$; $\quad S : \neg\Phi_{\mathcal{Q}} \rightsquigarrow \square$?

Проблема 3: в выводе может строиться резольвента дизъюнкта-запроса и склейки дизъюнкта-правила:



Эту проблему тоже можно решить, перестроив вывод:



Лемма об основных вычислениях (доказательство)

$S = \{D_1, \ldots, D_m, \neg\Phi_{\mathcal{Q}}\}$; $\quad S : \neg\Phi_{\mathcal{Q}} \rightsquigarrow \square$?

Проблема 4 (последняя): в выводе может строиться склейка запроса

Рассмотрим последнюю такую склейку:

$$\mathcal{Q}_1 = \neg(C_1 \,\&\, \ldots \,\&\, C_m \,\&\, A \,\&\, A) \to \mathcal{Q}_2 = \neg(C_1 \,\&\, \ldots \,\&\, C_m \,\&\, A) \to \ldots \to \square$$

$$\mathcal{R}_2 \;\cdots\; \mathcal{R}_k$$

Из входного вывода $\square$ из $\mathcal{Q}_2$ можно получить входной вывод $\mathfrak{D}'$ дизъюнкта $\square$ из $\neg A$, удалив

- изображённые атомы $C_1, \ldots, C_m$,
- все атомы остальных запросов, получающиеся из них переписыванием без изменений или применением правила резолюции,
- все правила, применявшиеся ко всем этим атомам для получения резольвент и
- все дублирующиеся запросы

Применив к $\mathcal{Q}_1$ правила, применявшиеся в $\mathfrak{D}'$, можно получить входной вывод $\mathcal{Q}_2$ из $\mathcal{Q}_1$ ▼

Лемма о подъёме вычисления

Для любых ХЛП $\mathcal{P}$, ответа к ней θ и запроса $\mathcal{Q}$, таких что $\mathcal{Q}\theta$ — основной запрос, верно: если для $[\mathcal{P}]$ существует успешное SLD-резолютивное вычисление, порождённое запросом $\mathcal{Q}\theta$, то для $\mathcal{P}$ существует успешное SLD-резолютивное вычисление, порождённое запросом $\mathcal{Q}$, результат которого — обобщение подстановки θ

Доказательство

Рассмотрим успешное SLD-резолютивное вычисление программы $[\mathcal{P}]$, порождённое запросом $\mathcal{Q}\theta$:

$$\mathcal{Q}_1^g \xrightarrow{\mathcal{R}_1^g, k_1, \varepsilon} \mathcal{Q}_2^g \xrightarrow{\mathcal{R}_2^g, k_2, \varepsilon} \ldots \xrightarrow{\mathcal{R}_n^g, k_n, \varepsilon} \square$$

Покажем, как по нему построить успешное SLD-резолютивное вычисление программы $\mathcal{P}$, порождённое запросом $\mathcal{Q}$:

$$\mathcal{Q}_1 \xrightarrow{\mathcal{R}_1, k_1, \eta_1} \mathcal{Q}_2 \xrightarrow{\mathcal{R}_2, k_2, \eta_2} \ldots \xrightarrow{\mathcal{R}_n, k_n, \eta_n} \square$$

Лемма о подъёме вычисления (доказательство)

$$\mathcal{Q}_1^g \xrightarrow{\mathcal{R}_1^g, k_1, \varepsilon} \mathcal{Q}_2^g \xrightarrow{\mathcal{R}_2^g, k_2, \varepsilon} \ldots \xrightarrow{\mathcal{R}_n^g, k_n, \varepsilon} \square$$

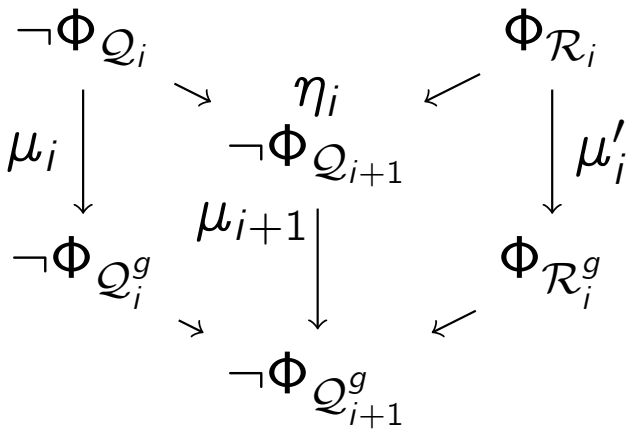
$$(\exists?)\ \mathcal{Q}_1 \xrightarrow{\mathcal{R}_1, k_1, \eta_1} \mathcal{Q}_2 \xrightarrow{\mathcal{R}_2, k_2, \eta_2} \ldots \xrightarrow{\mathcal{R}_n, k_n, \eta_n} \square$$

Чтобы «поднять» один шаг вывода, воспользуемся леммой о подъёме для дизъюнктов и соответствием вычислений ХЛП и входных выводов

$\neg\Phi_{\mathcal{Q}_i^g}$ и $\Phi_{\mathcal{R}_i^g}$ — это основные примеры дизъюнктов $\neg\Phi_{\mathcal{Q}_i}$ и $\Phi_{\mathcal{R}_i}$

$\neg\Phi_{\mathcal{Q}_{i+1}^g}$ — это резольвента дизъюнктов $\neg\Phi_{\mathcal{Q}_i^g}$ и $\Phi_{\mathcal{R}_i^g}$

Значит, согласно лемме о подъёме и её доказательству, соседние запросы вычислений соотносятся так:



Резольвента $\neg\Phi_{\mathcal{Q}_{i+1}}$ дизъюнктов $\neg\Phi_{\mathcal{Q}_i}$ и $\Phi_{\mathcal{R}_i}$ для унификатора η_i существует, и $\mu_i \cup \mu_i' = \eta_i\mu_{i+1}$, то есть $\mu_i = (\eta_i\mu_{i+1})|_{\mathrm{Var}\mathcal{Q}_i}$, и при этом $\mu_1 = \theta$

Последовательно применяя это соотношение к $i = 1, 2, \ldots$, получим требуемое вычисление, а также соотношение $\theta = (\eta_1\eta_2 \ldots \eta_n)|_{\mathrm{Var}_\mathcal{Q}}\mu$ для некоторой подстановки μ ▼

Теорема о полноте операционной семантики ХЛП

Для любых ХЛП $\mathcal{P}$ и запроса $\mathcal{Q}$ любой правильный ответ на запрос $\mathcal{Q}$ к $\mathcal{P}$ является частным случаем хотя бы одного SLD-вычислимого ответа на запрос $\mathcal{Q}$ к $\mathcal{P}$

Доказательство. Рассмотрим правильный ответ θ на запрос $\mathcal{Q}$ к $\mathcal{P}$

Пусть $\mathrm{Var}_{\mathcal{Q}\theta} = \{\mathrm{Z}_1, \ldots, \mathrm{Z}_k\}$

Выберем константы $\mathbf{c}_1, \ldots, \mathbf{c}_k$, не содержащиеся ни в $\mathcal{P}$, ни в $\mathcal{Q}$, ни в термах из θ

Символом μ обозначим подстановку $\{\mathrm{Z}_1/\mathbf{c}_1, \ldots, \mathrm{Z}_\mathrm{k}/\mathbf{c}_k\}$

θ — правильный ответ, а значит, верно соотношение $\mathcal{P} \models \forall\widetilde{\mathrm{Z}^\mathrm{k}}\ (\mathcal{Q}\theta)$

Следовательно, верно и $\mathcal{P} \models \mathcal{Q}\theta\mu$

По лемме об основных вычислениях, существует успешное SLD-резолютивное вычисление программы $[\mathcal{P}]$, порождённое запросом $\mathcal{Q}\theta\mu$

По лемме о подъёме вычисления, существует успешное SLD-резолютивное вычисление программы $\mathcal{P}$, порождённое запросом $\mathcal{Q}$, и для результата η этого вычисления и некоторой подстановки ρ верно $\theta\mu = \eta\rho$

Теорема о полноте операционной семантики ХЛП

Доказательство

$\mathrm{Var}_{\mathcal{Q}\theta} = \{\mathrm{Z}_1, \ldots, \mathrm{Z}_k\}$; $\mu = \{\mathrm{Z}_1/\mathbf{c}_1, \ldots, \mathrm{Z}_k/\mathbf{c}_k\}$; η — SLD-вычислимый ответ; $\exists \rho : \theta\mu = \eta\rho$

Заме́ним константы $\mathbf{c}_1, \ldots, \mathbf{c}_k$ на переменные $\mathrm{Z}_1, \ldots, \mathrm{Z}_k$ во всех термах в правых частях связок подстановок θ, μ, η и ρ

Так как константы $\mathbf{c}_1, \ldots, \mathbf{c}_k$ не содержатся в $\mathcal{P}$ и в $\mathcal{Q}$, то они не содержатся и в подстановке η

Подстановка μ в результате такой замены заменится на ε

Значит, в результате такой замены равенство

$$\theta\mu = \eta\rho$$

преобразуется в

$$\theta = \eta\rho'$$

для некоторой подстановки ρ', то есть правильный ответ θ действительно является частным случаем некоторого SLD-вычислимого ответа η ▼

Математическая логика и логическое программирование

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

Блок 36

Хорновские логические программы:
содержательное сравнение семантик

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

ВМК МГУ, 2023/2024, осенний семестр

Для ХЛП были введены две семантики:

- Декларативная:
  - программа и запрос — это формулы
  - правильный ответ — это такой, для которого запрос логически следует из программы
- Операционная:
  - шаг вычисления — это построение SLD-резольвенты
  - SLD-вычислимый ответ — это такой, который можно получить при помощи последовательности таких шагов

С одной стороны, операционная семантика может расцениваться как «чисто техническое» дополнение к декларативной, в котором говорится, как именно на компьютере вычисляются правильные ответы

С другой стороны, так как вычисление программы на компьютере — это важно и неизбежно, то хотелось бы иметь более «интуитивно ясное» понимание происходящего

Запрос «$?C_1, \ldots, C_k$» — это:

- Согласно декларативной семантике: обращённый к программе вопрос «для каких значений целевых переменных утверждения $C_1, \ldots C_k$ обязательно верны, если считать верным всё то, что записано в программе?»
- Согласно операционной семантике: список задач, записанных в виде атомов, общий ответ к которым нас интересует

Факт «$A;$» — это:

- Согласно декларативной семантике: утверждение о том, что A безусловно верно (*для любых значений используемых переменных*)
- Согласно операционной семантике: тривиальный способ решения задачи A, в котором сразу выдаётся ответ
  - Правило SLD-резолюции — это объявление об успешном решении задачи A с записью ответа в унификатор

Правило «$A \leftarrow B_1, \ldots, B_k$» для $k \geq 1$ — это:

- Согласно декларативной семантике: утверждение о том, что если верны все утверждения $B_1, \ldots, B_k$, то верно и A
- Согласно операционной семантике: запись способа решения задачи A, согласно которому следует решить (под)задачи $B_1, \ldots, B_k$ и скомбинировать ответ к A из ответов к этим подзадачам
  - Правило SLD-резолюции — переход к решению подзадач с записью способа комбинирования ответов в унификатор

ХЛП $\mathcal{P}$ — это:

- Согласно декларативной семантике: база знаний, посчитанных правильными, существенными и достаточными для ответов на интересующие вопросы
- Согласно операционной семантике: запись всевозможных способов решения (достаточного набора решений) интересующих задач и всех возникающих подзадач

С точки зрения декларативной семантики, порядок записи утверждений в программе и атомов в телах правил и запросов неважны:

- Конъюнкция коммутативна (можно переставлять атомы в телах без изменения смысла формул)
- Множество — это неупорядоченная совокупность элементов

С точки зрения операционной семантики, резонно предположение, что порядок записи утверждений и атомов в телах может быть важен:

- Зависят ли результат вычисления от выбора конкретных способов решения подзадач?
  - *Очевидно, что да*: ранее уже были примеры вычислений, порождённых одним и тем же запросом к одной и той же программе, но приводящих к разным ответам или вовсе не приводящих к ответу
- Зависит ли результат вычисления от выбора порядка решения подзадач?
  - *А это не так очевидно*

Математическая логика и логическое программирование

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

Блок 37

Хорновские логические программы:
переключательная лемма,
сильная полнота операционной семантики,
стандартное правило выбора подцели

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

$$\mathcal{R}_1 : \mathrm{p}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}); \quad \mathcal{R}_2 : \mathrm{q}(\mathbf{b}); \quad \mathcal{R}_3 : \mathrm{r}(\mathbf{c}); \quad \mathcal{R}_4 : \mathrm{s}(\mathbf{b});$$

Запросом $?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$ к этой программе порождаются, например, такие SLD-резолютивные вычисления, различающиеся тем, что в одном всегда выбирается первая подцель, а в другом — последняя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$
$\mathcal{R}'_1, 1 \downarrow \theta_1 = \{\mathtt{X}'/\mathtt{X}, \mathtt{Y}'/\mathtt{Y}\}$
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$
$\mathcal{R}'_2, 1 \downarrow \theta_2 = \{\mathtt{X}/\mathbf{b}\}$
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$
$\mathcal{R}'_3, 1 \downarrow \theta_3 = \{\mathtt{Y}/\mathbf{c}\}$
$?\mathrm{s}(\mathbf{b})$
$\mathcal{R}'_4, 1 \downarrow \theta_4 = \varepsilon$
$\square$

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$
$\mathcal{R}'_4, 2 \downarrow \eta_1 = \{\mathtt{X}/\mathbf{b}\}$
$?\mathrm{p}(\mathbf{b}, \mathtt{Y})$
$\mathcal{R}'_1, 1 \downarrow \eta_2 = \{\mathtt{X}'/\mathbf{b}, \mathtt{Y}'/\mathtt{Y}\}$
$?\mathrm{q}(\mathbf{b}), \mathrm{r}(\mathtt{Y})$
$\mathcal{R}'_3, 2 \downarrow \eta_3 = \{\mathtt{Y}/\mathbf{c}\}$
$?\mathrm{q}(\mathbf{b})$
$\mathcal{R}'_2, 1 \downarrow \eta_4 = \varepsilon$
$\square$

Результаты этих вычислений:

$(\theta_1\theta_2\theta_3\theta_4)|_{\{\mathtt{X},\mathtt{Y}\}} = \{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{c}\}, \qquad (\eta_1\eta_2\eta_3\eta_4)|_{\{\mathtt{X},\mathtt{Y}\}} = \{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{c}\}$ — оказались одинаковыми

А это случайное совпадение, или так бывает всегда?

Переключательная лемма

Для любого SLD-резолютивного вычисления

$$\mathcal{Q} = ?C_1, \ldots, C_{i-1}, C_i, C_{i+1}, \ldots, C_{j-1}, C_j, C_{j+1}, \ldots, C_m$$
$$\mathcal{R}\downarrow\theta_1$$
$$?(C_1, \ldots, C_{i-1}, \mathcal{R}^-, C_{i+1}, \ldots, C_{j-1}, C_j, C_{j+1}, \ldots, C_m)\theta_1$$
$$\mathcal{S}\downarrow\theta_2$$
$$\mathcal{Q}_1 = ?(C_1, \ldots, C_{i-1}, \mathcal{R}^-, C_{i+1}, \ldots, C_{j-1})\theta_1\theta_2, \mathcal{S}^-\theta_2, (C_{j+1}, \ldots, C_m)\theta_1\theta_2$$

существует SLD-резолютивное вычисление

$$\mathcal{Q} = ?C_1, \ldots, C_{i-1}, C_i, C_{i+1}, \ldots, C_{j-1}, C_j, C_{j+1}, \ldots, C_m$$
$$\widetilde{\mathcal{S}}\downarrow\eta_1$$
$$?(C_1, \ldots, C_{i-1}, C_i, C_{i+1}, \ldots, C_{j-1}, \widetilde{\mathcal{S}}^-, C_{j+1}, \ldots, C_m)\eta_1$$
$$\widetilde{\mathcal{R}}\downarrow\eta_2$$
$$\mathcal{Q}_2 = ?(C_1, \ldots, C_{i-1})\eta_1\eta_2, \widetilde{\mathcal{R}}^-\eta_2, (C_{i+1}, \ldots, C_{j-1}, \widetilde{\mathcal{S}}^-, C_{j+1}, \ldots, C_m)\eta_1\eta_2,$$

такое что

- $\widetilde{\mathcal{R}}$ **и** $\widetilde{\mathcal{S}}$ **— варианты правил** $\mathcal{R}$ **и** $\mathcal{S}$ **соответственно и**
- $\mathcal{Q}_1 = \mathcal{Q}_2$ **и** $\theta_1\theta_2|_{\mathrm{Var}_\mathcal{Q}} = \eta_1\eta_2|_{\mathrm{Var}_\mathcal{Q}}$

как для соотношения $i < j$**, так и для** $j < i$

($\mathcal{R}^+$ и $\mathcal{R}^-$ — так будем обозначать соответственно заголовок и тело правила $\mathcal{R}$)

Переключательная лемма (доказательство)

Для технической простоты положим, что $i = 1$ и $j = m = 2$ (это не повлияет на общих ход рассуждений, но упростит иллюстрации)

То есть покажем, что для любого SLD-резолютивного вычисления

$$?A, B \xrightarrow{\mathcal{R},\theta_1} ?\mathcal{R}^-\theta_1, B\theta_1 \xrightarrow{\mathcal{S},\theta_2} \mathcal{Q}_1 = ?\mathcal{R}^-\theta_1\theta_2, \mathcal{S}^-\theta_2$$

существует SLD-резолютивное вычисление

$$?A, B \xrightarrow{\widetilde{\mathcal{S}},\eta_1} ?A\eta_1, \widetilde{\mathcal{S}}^-\eta_1 \xrightarrow{\widetilde{\mathcal{R}},\eta_2} \mathcal{Q}_2 = ?\widetilde{\mathcal{R}}^-\eta_2, \widetilde{\mathcal{S}}^-\eta_1\eta_2,$$

такое что $\widetilde{\mathcal{R}}$ и $\widetilde{\mathcal{S}}$ — варианты правил $\mathcal{R}$ и $\mathcal{S}$ и $\mathcal{Q}_2 = \mathcal{Q}_1$

Рассмотрим произвольный вариант $\widetilde{\mathcal{S}}$ правила $\mathcal{S}$ ($\mathcal{S} = \widetilde{\mathcal{S}}\mu$), в котором нет переменных из $\mathrm{Var}_B \cup \mathrm{Var}_{B\theta_1} \cup \mathrm{Var}_{\mathcal{R}} \cup \mathrm{Var}_{\mathcal{R}^-\theta_1} \cup \mathrm{Dom}_{\theta_1}$

Так как $(\mathrm{Var}_{B\theta_1} \cup \mathrm{Var}_{\mathcal{R}^-\theta_1}) \cap (\mathrm{Var}_{\mathcal{S}} \cup \mathrm{Var}_{\widetilde{\mathcal{S}}}) = \emptyset$, то можно выбрать такое переименование μ, чтобы было верно $\mathrm{Dom}_\mu \cap (\mathrm{Var}_{B\theta_1} \cup \mathrm{Var}_{\mathcal{R}^-\theta_1}) = \emptyset$

По условию верно $(B\theta_1)\theta_2 = \mathcal{S}^+\theta_2$

Следовательно, $B\theta_1\mu\theta_2 = B\theta_1\theta_2 = \mathcal{S}^+\theta_2 = \widetilde{\mathcal{S}}^+\mu\theta_2 = \widetilde{\mathcal{S}}^+\theta_1\mu\theta_2$

Значит, $\theta_1\mu\theta_2$ — унификатор атомов B и $\widetilde{\mathcal{S}}^+$

Переключательная лемма (доказательство)

$$?A, B \xrightarrow{\mathcal{R},\theta_1} ?\mathcal{R}^-\theta_1, B\theta_1 \xrightarrow{\mathcal{S},\theta_2} \mathcal{Q}_1 = ?\mathcal{R}^-\theta_1\theta_2, \mathcal{S}^-\theta_2$$

$$\exists ?\ ?A, B \xrightarrow{\widetilde{\mathcal{S}},\eta_1} ?A\eta_1, \widetilde{\mathcal{S}}^-\eta_1 \xrightarrow{\widetilde{\mathcal{R}},\eta_2} \mathcal{Q}_2 = ?\widetilde{\mathcal{R}}^-\eta_2, \widetilde{\mathcal{S}}^-\eta_1\eta_2$$

$$\mathcal{S} = \widetilde{\mathcal{S}}\mu \qquad B\theta_1\mu\theta_2 = \widetilde{\mathcal{S}}^+\theta_1\mu\theta_2$$

По следствию из теоремы об унификации, существует и наиболее общий унификатор η_1 этих атомов ($\theta_1\mu\theta_2 = \eta_1\rho$), и для него

$$?A, B \xrightarrow{\widetilde{\mathcal{S}},\eta_1} ?A\eta_1, \widetilde{\mathcal{S}}^-\eta_1$$

Так как $(\mathrm{Var}_B \cup \mathrm{Var}_{\widetilde{\mathcal{S}}}) \cap \mathrm{Var}_{\mathcal{R}} = \emptyset$, то, используя переименование переменных, можно выбрать такой наиболее общий унификатор η_1, для которого верно $\mathrm{Dom}_{\eta_1} \cap \mathrm{Var}_{\mathcal{R}} = \emptyset$

По условию верно $A\theta_1 = \mathcal{R}^+\theta_1$

Следовательно, верно и $A\eta_1\rho = A\theta_1\mu\theta_2 = \mathcal{R}^+\theta_1\mu\theta_2 = \mathcal{R}^+\eta_1\rho = \mathcal{R}^+\rho$

Значит, ρ — унификатор атомов $A\eta_1$ и $\mathcal{R}^+$, и существует их наиболее общий унификатор η_2 ($\rho = \eta_2\rho'$), и для него

$$?A, B \xrightarrow{\widetilde{\mathcal{S}},\eta_1} ?A\eta_1, \widetilde{\mathcal{S}}^-\eta_1 \xrightarrow{\mathcal{R},\eta_2} ?\mathcal{R}^-\eta_2, \widetilde{\mathcal{S}}^-\eta_1\eta_2$$

Переключательная лемма (доказательство)

$$?A, B \xrightarrow{\mathcal{R},\theta_1} ?\mathcal{R}^-\theta_1, B\theta_1 \xrightarrow{\mathcal{S},\theta_2} \mathcal{Q}_1 = ?\mathcal{R}^-\theta_1\theta_2, \mathcal{S}^-\theta_2$$

$$\exists\ ?A, B \xrightarrow{\widetilde{\mathcal{S}},\eta_1} ?A\eta_1, \widetilde{\mathcal{S}}^-\eta_1 \xrightarrow{\mathcal{R},\eta_2} \mathcal{Q}_2 = ?\mathcal{R}^-\eta_2, \widetilde{\mathcal{S}}^-\eta_1\eta_2$$

$$\mathcal{S} = \widetilde{\mathcal{S}}\mu \qquad B\theta_1\mu\theta_2 = \widetilde{\mathcal{S}}^+\theta_1\mu\theta_2$$

$$(A\eta_1)\eta_2\rho' = \mathcal{R}^+\eta_2\rho' \qquad \theta_1\mu\theta_2 = \eta_1\eta_2\rho'$$

При этом $\mathcal{R}^-\eta_1 = \mathcal{R}^-$, а значит,
$\mathcal{R}^-\theta_1\theta_2 = \mathcal{R}^-\theta_1\mu\theta_2 = \mathcal{R}^-\eta_1\eta_2\rho' = \mathcal{R}^-\eta_2\rho'$, то есть атом $\mathcal{R}^-\theta_1\theta_2$ — пример атома $\mathcal{R}^-\eta_2$

Кроме того, $\widetilde{\mathcal{S}}^-\eta_1\eta_2\rho' = \widetilde{\mathcal{S}}^-\theta_1\mu\theta_2 = \widetilde{\mathcal{S}}^-\mu\theta_2 = \mathcal{S}^-\theta_2$, то есть атом $\mathcal{S}^-\theta_2$ — пример атома $\widetilde{\mathcal{S}}^-\eta_1\eta_2$

Применяя те же рассуждения о взаимосвязи унификаторов от нижнего вычисления к верхнему, можно получить и обратное: $\mathcal{R}^-\eta_2$ и $\widetilde{\mathcal{S}}^-\eta_1\eta_2\rho$ — примеры атомов $\mathcal{R}^-\theta_1\theta_2$ и $\mathcal{S}^-\theta_2$ соответственно

Если атомы являются примерами друг друга, то они (очевидно, что) являются вариантами друг друга, а значит, $\mathcal{Q}_1$ — вариант $\mathcal{Q}_2$:
$\mathcal{Q}_1 = \mathcal{Q}_2\mu'$ для некоторого переименования μ'

Осталось заменить в нижнем вычислении унификатор η_2 на $\eta_2\mu'$ ▼

Сильная полнота операционной семантики

Правило выбора подцели — это отображение $\mathfrak{R}$, сопоставляющее каждому неуспешному вычислению $\mathfrak{S}$ номер подцели $\mathfrak{R}(\mathfrak{S})$ его последнего запроса

$\mathfrak{R}$-вычисление — это SLD-резолютивное вычисление, в котором для каждого префикса $\mathfrak{S} \to \mathcal{Q}$ резольвента $\mathcal{Q}$ строится для $\mathfrak{R}(\mathfrak{S})$-й подцели

Результаты $\mathfrak{R}$-вычислений называются $\mathfrak{R}$-вычислимыми ответами

Теорема (о сильной полноте операционной семантики ХЛП). Для любого правила выбора подцели $\mathfrak{R}$ и любого правильного ответа θ существует $\mathfrak{R}$-вычислимый ответ, являющийся обобщением θ

Доказательство.

По теореме о полноте операционной семантики ХЛП, существует SLD-вычислимый ответ η, являющийся обобщением θ

По определению SLD-вычислимого ответа, существует успешное вычисление с результатом η:

$$\mathcal{Q}_1 \xrightarrow{\mathcal{R}_1, k_1, \eta_1} \mathcal{Q}_2 \xrightarrow{\mathcal{R}_2, k_2, \eta_2} \ldots \xrightarrow{\mathcal{R}_n, k_n, \eta_n} \square$$

Сильная полнота операционной семантики

Доказательство.

$$\mathfrak{D} = (\mathcal{Q}_1 \xrightarrow{\mathcal{R}_1, k_1, \eta_1} \mathcal{Q}_2 \xrightarrow{\mathcal{R}_2, k_2, \eta_2} \ldots \xrightarrow{\mathcal{R}_n, k_n, \eta_n} \square)$$

Если $\mathfrak{D}$ является $\mathfrak{R}$-вычислением, то теорема доказана: η — $\mathfrak{R}$-вычислимый ответ, являющийся обобщением θ

Иначе в $\mathfrak{D}$ существует шаг i, такой что $\mathfrak{R}(\mathcal{Q}_1, \ldots, \mathcal{Q}_i) = m \neq k_i$

Так как $\mathfrak{D}$ завершается пустым запросом, то на некотором шаге j, $j > i$, резольвента строится для подцели, отвечающей m-й подцели запроса $\mathcal{Q}_i$

Используя переключательную лемму не более n раз, можно переупорядочить шаги вычисления от i-го до j-го так, чтобы

- на i-м шаге резольвента строилась для m-й подцели,
- запрос $\mathcal{Q}_{j+1}$ остался прежним и
- значение $(\eta_i \ldots \eta_j)|_{\mathrm{Var}_{\mathcal{Q}_i}}$ осталось прежним

Применяя такое переупорядочивание не более n раз, можно последовательно устранить все шаги вычисления в $\mathfrak{D}$, на которых подцель выбирается «неправильно» относительно $\mathfrak{R}$ и тем самым перестроить $\mathfrak{D}$ в успешное $\mathfrak{R}$-вычисление с тем же результатом ▼

Стандартное правило выбора подцели

Согласно теореме о сильной полноте операционной семантики ХЛП, независимо от того, какое именно используется правило выбора подцели $\mathfrak{R}$, всегда есть возможность, придерживаясь правила $\mathfrak{R}$, получить всевозможные SLD-вычислимые ответы

Чтобы упростить анализ и использование ХЛП, как на практике, так и в теории зачастую используется стандартное правило выбора подцели: в каждом запросе всегда выбирается первая (самая левая) подцель

В изображении способа получения SLD-резольвенты $\mathcal{Q}_1 \xrightarrow{\mathcal{R},1,\theta} \mathcal{Q}_2$ согласно стандартному правилу номер подцели (1) будет опускаться: $\mathcal{Q}_1 \xrightarrow{\mathcal{R},\theta} \mathcal{Q}_2$

Математическая логика и логическое программирование

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

Блок 38

Хорновские логические программы:
деревья SLD-резолютивных вычислений,
стратегии вычисления и их полнота,
стандартная стратегия вычисления

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

$$\mathcal{R}_1: \ \mathrm{p}(\mathtt{X},\mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$\mathcal{R}_2: \ \mathrm{p}(\mathtt{X},\mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$\mathcal{R}_3: \mathrm{q}(\mathbf{b}); \quad \mathcal{R}_4: \mathrm{r}(\mathbf{c}); \quad \mathcal{R}_5: \mathrm{s}(\mathbf{b});$$

Запросом $?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$ к этой программе порождаются, например, такие SLD-резолютивные вычисления, построенные согласно стандартному правилу выбора подцели:

$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$

$\mathcal{R}'_1 \downarrow \theta_1 = \{\mathtt{X}'/\mathtt{X}, \mathtt{Y}'/\mathtt{Y}\}$

$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$

$\mathcal{R}'_3 \downarrow \theta_2 = \{\mathtt{X}/\mathbf{b}\}$

$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$

$\mathcal{R}'_4 \downarrow \theta_3 = \{\mathtt{Y}/\mathbf{c}\}$

$?\mathrm{s}(\mathbf{b})$

$\mathcal{R}'_5 \downarrow \theta_4 = \varepsilon$

$\square$

$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$

$\mathcal{R}'_2 \downarrow \eta_1 = \{\mathtt{X}'/\mathtt{X}, \mathtt{Y}/\mathtt{X}\}$

$?\mathrm{r}(\mathtt{X}), \mathrm{s}(\mathtt{X})$

$\mathcal{R}'_4 \downarrow \eta_2 = \{\mathtt{X}/\mathbf{c}\}$

$?\mathrm{s}(\mathbf{c})$

тупик

Вступление

Способ выбора правил программы при построении SLD-резолютивных вычислений существенно влияет на то, какой именно результат будет получен (и будет ли получен хоть какой-нибудь результат)

Этот факт нетруден для осознания:

- Правило — это описание одного из способов решения рассматриваемой задачи
- Бывают как успешные, так и неуспешные способы решения задач
  - Например, если заданный элемент списка располагается только в голове, то искать его в хвосте — заведомый неуспех
- Задачу можно успешно решать по-разному, получая разные ответы
  - Например, если в ответе требуется произвольный элемент списка, то можно выдать голову списка, или голову его хвоста, или голову хвоста его хвоста, ...

Чтобы умело рассуждать о способах выбора правил для получения ответов, объединим всевозможные SLD-резолютивные вычисления программ в единую удобную структуру

Деревья SLD-резолютивных вычислений ХЛП

Дерево SLD-резолютивных вычислений для запроса $\mathcal{Q}$ к программе $\mathcal{P}$ и правила выбора подцели $\mathfrak{R}$, — это размеченное корневое ориентированное (возможно, бесконечное) дерево $T^{\mathfrak{R}}_{\mathcal{P},\mathcal{Q}}$, устроенное так:

1. Каждая вершина помечена запросом
2. Корень помечен запросом $\mathcal{Q}$
3. Дýги имеют вид $\boxed{\mathcal{Q}_1} \xrightarrow{\mathcal{R},\theta} \boxed{\mathcal{Q}_2}$:
   - $\mathcal{Q}_2$ — SLD-резольвента $\mathcal{Q}_1$ и утверждения из $\mathcal{P}$ с вариантом $\mathcal{R}$ и унификатором θ
   - Резольвента строится для подцели $\mathfrak{R}(\mathfrak{S})$, где $\mathfrak{S}$ — вычисление от корня до изображённой вершины $\boxed{\mathcal{Q}_1}$
   - Дуги из $\boxed{\mathcal{Q}_1}$ отвечают выбору всевозможных правил из $\mathcal{P}$ для построения резольвент с $\mathcal{Q}_1$ и упорядочены по порядку записи правил в $\mathcal{P}$
4. Листьями являются запросы, у которых нет ни одной SLD-резольвенты правилами из $\mathcal{P}$

$T_{\mathcal{P},\mathcal{Q}}$ — дерево $T^{\mathfrak{R}}_{\mathcal{P},\mathcal{Q}}$ для стандартного правила $\mathfrak{R}$

Порядок дуг дерева в иллюстрациях — слева направо

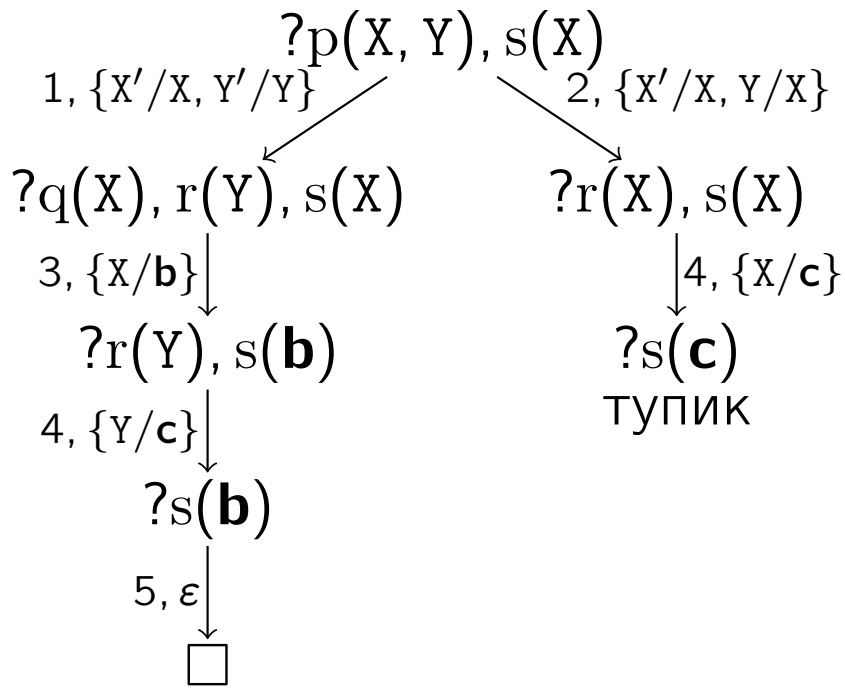
Деревья SLD-резолютивных вычислений ХЛП

Примеры

Программа $\mathcal{P}$:

$$1: \ \mathrm{p}(\mathtt{X},\mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$2: \ \mathrm{p}(\mathtt{X},\mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$3: \mathrm{q}(\mathbf{b}); \quad 4: \mathrm{r}(\mathbf{c}); \quad 5: \ \mathrm{s}(\mathbf{b});$$

Дерево $T_{\mathcal{P},?\mathrm{p}(\mathtt{X},\mathtt{Y}),\mathrm{s}(\mathtt{X})}$:



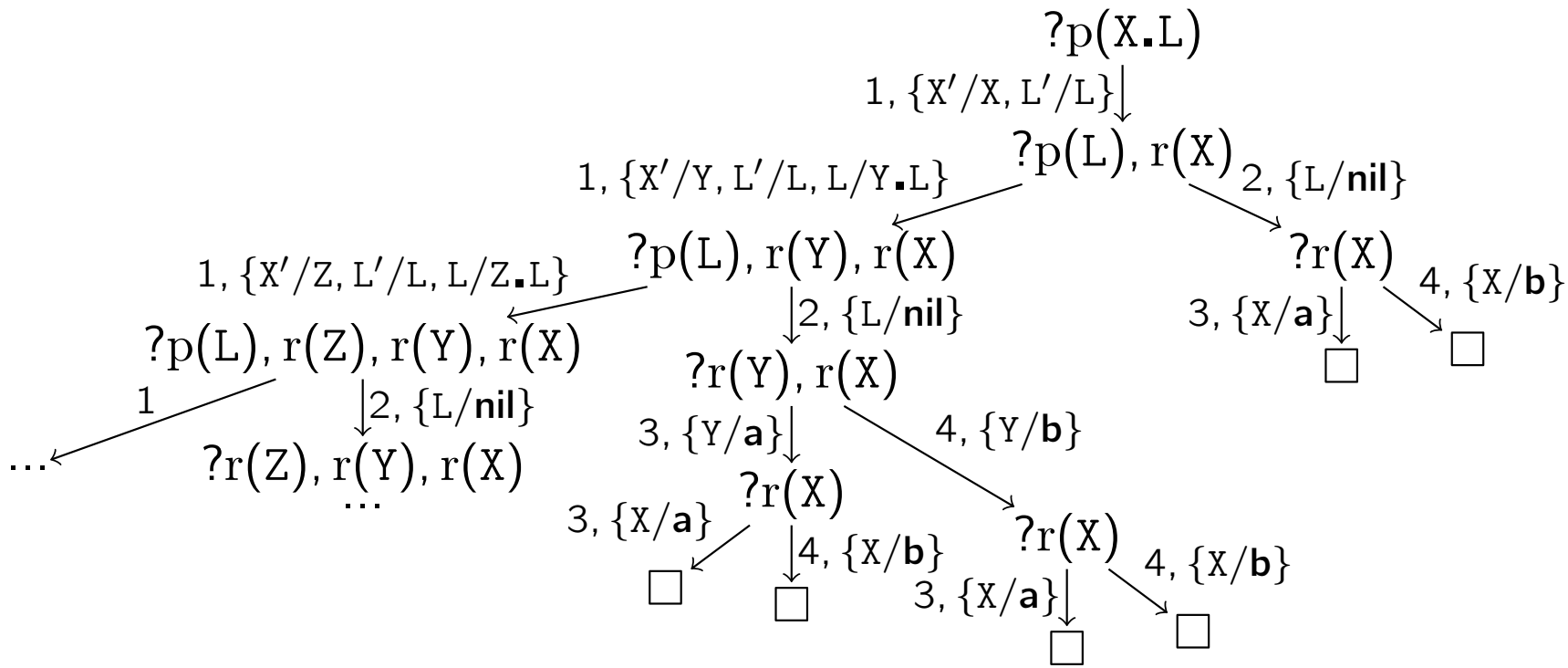
Деревья SLD-резолютивных вычислений ХЛП

Примеры

Программа $\mathcal{P}$:

$$1: \ \mathrm{p}(\mathrm{X}.\mathrm{L}) \leftarrow \mathrm{p}(\mathrm{L}), \mathrm{r}(\mathrm{X});$$
$$2: \mathrm{p}(\mathbf{nil}); \quad 3: \mathrm{r}(\mathbf{a}); \quad 4: \mathrm{r}(\mathbf{b});$$

Дерево $T_{\mathcal{P},?\mathrm{p}(\mathrm{X}.\mathrm{L})}$:



Стратегии вычисления ХЛП

Деревья вычислений бывают разные:

- Конечные и бесконечные
- С конечным и с бесконечным числом ветвей
- Всюду успешные и с тупиками
- Содержащие один ответ, или много ответов, или ни одного ответа

Каждая ветвь в дереве $T^{\mathfrak{R}}_{\mathcal{Q},\mathcal{P}}$ отвечает SLD-резолютивному вычислению программы $\mathcal{P}$ для запроса $\mathcal{Q}$ согласно правилу $\mathfrak{R}$, и в дереве перечислены все такие вычисления

Значит, перебор всех таких вычислений можно устроить как **обход** дерева $T^{\mathfrak{R}}_{\mathcal{Q},\mathcal{P}}$

Стратегии вычисления ХЛП

Стратегия вычисления ХЛП состоит из

- правила выбора подцели и
- способа обхода дерева вычислений

Стратегия вычисления с правилом выбора подцели $\mathfrak{R}$ называется полной, для любого запроса $\mathcal{Q}$ к любой программе $\mathcal{P}$ она позволяет построить (перечислить) все успешные $\mathfrak{R}$-вычисления $\mathcal{P}$ для запроса $\mathcal{Q}$

Деревом SLD-резолютивных вычислений для запроса $\mathcal{Q}$ к программе $\mathcal{P}$, построенным согласно заданной стратегии вычисления, будем называть фрагмент дерева SLD-резолютивных вычислений для $\mathcal{Q}$, $\mathcal{P}$ и правила выбора подцели из стратегии, состоящий из всех вершин, посещаемых при обходе, и всех соединяющих их дуг

Стратегии вычисления ХЛП

Два основных вида обходов деревьев вычислений ХЛП:

- Обход в ширину: вершины дерева обходятся поярусно по неубыванию удалённости от корня
  - Это можно представить как перебор всех вычислений по возрастанию длины

- Обход в глубину: при обходе из вершины сначала последовательно по порядку обходятся вершины по исходящим дугам, и по окончании обхода выполняется возврат в предыдущую вершину
  - Это можно представить себе как последовательные попытки достроить (одно) рассматриваемое вычисление до более длинного пошагово всеми возможными способами по порядку правил

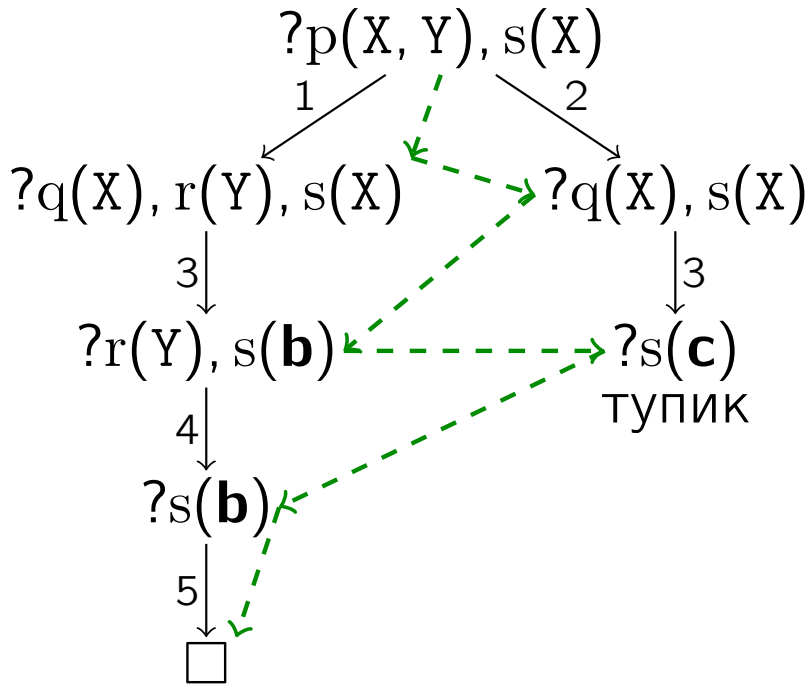
Стратегии вычисления ХЛП

Пример дерева вычислений, построенного согласно стандартному правилу выбора подцели и обходу в ширину

Программа $\mathcal{P}$:

$$1: \ \mathrm{p}(\mathtt{X},\mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$2: \ \mathrm{p}(\mathtt{X},\mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$3: \mathrm{q}(\mathbf{b}); \quad 4: \mathrm{r}(\mathbf{c}); \quad 5: \quad \mathrm{s}(\mathbf{b});$$

Дерево вычислений (зелёным пунктиром обозначен порядок обхода вершин дерева):

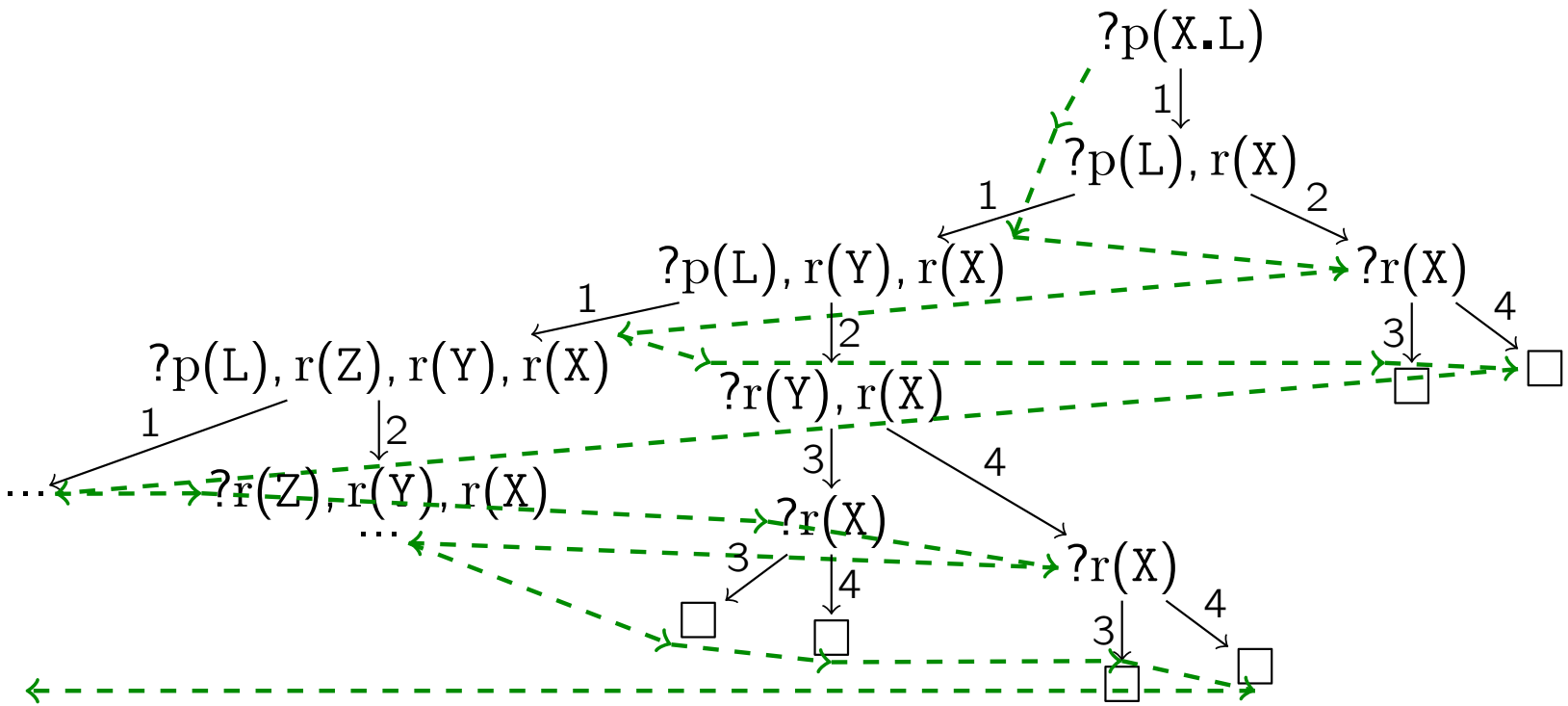


Стратегии вычисления ХЛП

Пример дерева вычислений, построенного согласно стандартному правилу выбора подцели и обходу в ширину

$$1: \quad \mathrm{p}(\mathtt{X.L}) \leftarrow \mathrm{p}(\mathtt{L}), \mathrm{r}(\mathtt{X});$$
$$2: \quad \mathrm{p}(\mathbf{nil}); \quad 3: \quad \mathrm{r}(\mathbf{a}); \quad 4: \quad \mathrm{r}(\mathbf{b});$$

Дерево $T_{\mathcal{P},?\mathrm{p}(\mathtt{X.L})}$:



Стратегии вычисления ХЛП

Стратегия обхода в ширину полна:

- Из каждой вершины исходит конечное число дуг (не больше чем правил в программе), а значит, каждый ярус дерева конечен
- Каждая вершина каждого яруса рано или поздно будет посещена
- Каждое успешное вычисление конечно, а значит, завершается на некотором ярусе
- Значит, каждое успешное вычисление рано или поздно будет построено

Но у этой стратегии есть и серьёзный недостаток, присущий обходам больших графов в ширину:

- При обходе нужно хранить в памяти **все** вершины очередного яруса
- Число таких вершин может расти экспоненциально относительно номера яруса
  - *можете посчитать, на каком ярусе двоичного дерева или, например, дерева со степенью ветвления 4 содержится гугóл вершин*

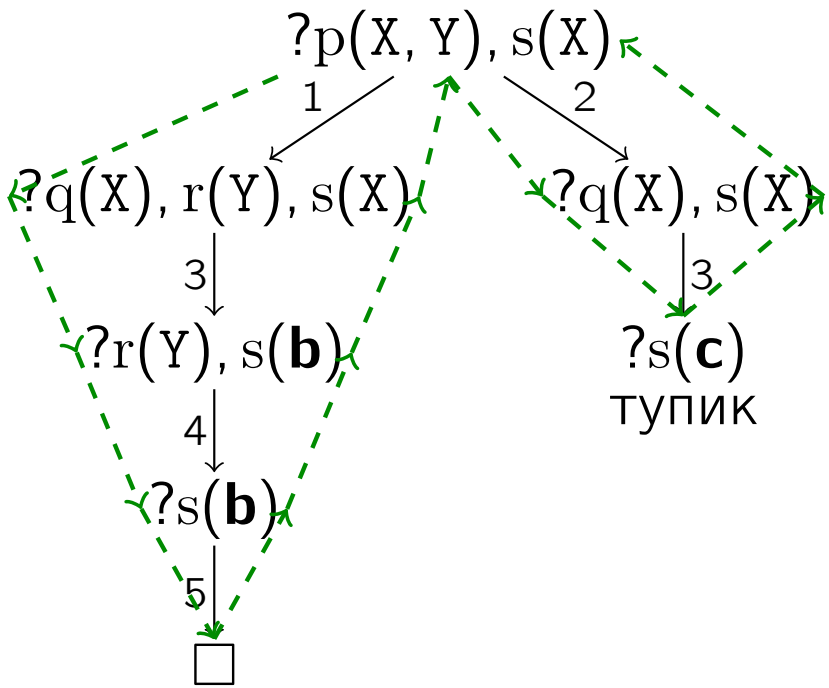
Стратегии вычисления ХЛП

Пример дерева вычислений, построенного согласно стандартному правилу выбора подцели и обходу в глубину

Программа $\mathcal{P}$:

$$1: \ \mathrm{p}(\mathtt{X},\mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$2: \ \mathrm{p}(\mathtt{X},\mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$3: \mathrm{q}(\mathbf{b}); \quad 4: \mathrm{r}(\mathbf{c}); \quad 5: \ \mathrm{s}(\mathbf{b});$$

Дерево вычислений:

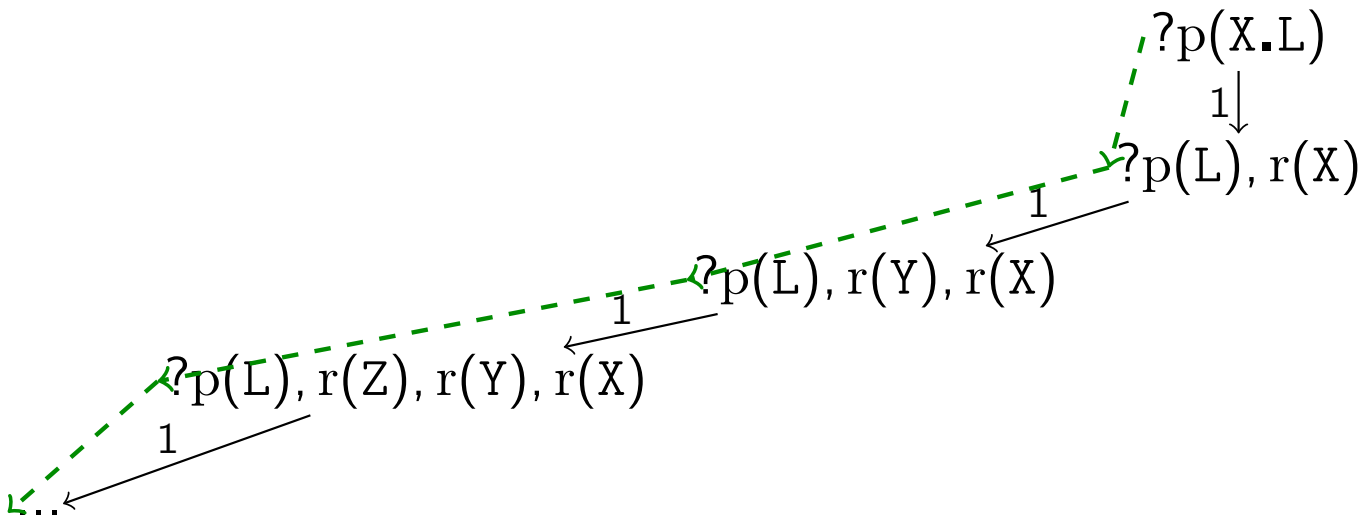


Стратегии вычисления ХЛП

Пример дерева вычислений, построенного согласно стандартному правилу выбора подцели и обходу в глубину

$$1: \ \mathrm{p}(\mathtt{X.L}) \leftarrow \mathrm{p}(\mathtt{L}), \mathrm{r}(\mathtt{X});$$
$$2: \mathrm{p}(\textbf{nil}); \quad 3: \mathrm{r}(\textbf{a}); \quad 4: \mathrm{r}(\textbf{b});$$

Дерево вычислений:



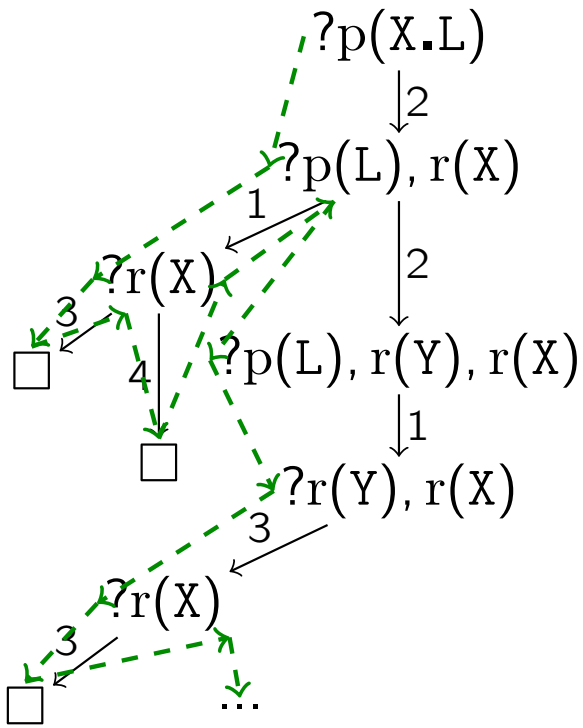
До бесконечности обходится самая левая ветвь дерева, ни одно успешное вычисление не найдено

Стратегии вычисления ХЛП

Пример дерева вычислений, построенного согласно стандартному правилу выбора подцели и обходу в глубину

$$1 : \mathrm{p}(\mathbf{nil});$$
$$2 : \ \mathrm{p}(\mathrm{X}.\mathrm{L}) \leftarrow \mathrm{p}(\mathrm{L}), \mathrm{r}(\mathrm{X});$$
$$3 : \mathrm{r}(\mathbf{a}); \qquad 4 : \mathrm{r}(\mathbf{b});$$

Дерево вычислений:



Стандартная стратегия вычисления ХЛП

По этим примерам видно, что стратегия, основанная на обходе в глубину

- неполна и
- чувствительна к порядку правил: даже если успешных вычислений сколь угодно много, в зависимости от порядка правил могут быть
  - найдены они все, или
  - найдены некоторые успешные вычисления, но не все, или
  - не найдены никакие успешные вычисления

Но использование этой стратегии позволяет избежать проблем с использованием памяти, возникающих при обходе в ширину, и она удобна и достаточно эффективна для использования на практике

Поэтому в стандартной стратегии вычисления ХЛП, используемой обычно в интерпретаторах, используются стандартное правило выбора подцели и обход в глубину

Выбор надлежащего порядка правил и атомов в телах правил при этом становится задачей программиста

Математическая логика и логическое программирование

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

Блок 39

Машины Тьюринга

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Хорновские логические программы — это вычислительная модель, основанная на проверке невыполнимости систем дизъюнктов частного вида (хорновских дизъюнктов) при помощи одного частного способа построения резолютивного вывода (входная резолюция)

При обсуждении полноты операционной семантики ХЛП было показано, что входная резолюция полна для систем хорновских дизъюнктов, но этими дизъюнктами выражаются только знания, «уложенные» в одну из двух частных строгих форм:

1. A;: утверждение, записанное как атом A, верно
2. $A \leftarrow B_1, \ldots, B_k$;: если утверждения, записанные как атомы $B_1, \ldots, B_k$, верны, то верно и утверждение, записанное как атом A

А не оказываются ли ХЛП слишком невыразительными из-за ограничений на форму представления знаний?

Какие задачи можно решить с помощью ХЛП, а какие нельзя?

Вступление

Есть много разных вычислительных моделей, предназначенных для работы с данными принципиального разного устройства: строками, числами, графами, типами, молекулами ДНК, *запросами к ХЛП*, ...

Естественно возникает потребность **сравнивать** между собой такие модели независимо от природы их данных

Естественный способ сравнения вычислительных моделей основан на том, что их данными кодируются неотрицательные целые числа ($\mathbb{N}_0$), и сравниваются возможности вычисления функций вида $f : \mathbb{N}_0 \to \mathbb{N}_0$

Тезис Чёрча-Тьюринга: все (частичные) функции вида $f : \mathbb{N}_0 \to \mathbb{N}_0$, которые вычисляются алгоритмами в нестрогом понимании, — это в точности все функции, вычислимые на **машинах Тьюринга**

Вступление

Тезис Чёрча-Тьюринга невозможно сформулировать строго и тем более обосновать, но подтверждением можно считать то, что многие модели, придумывавшиеся как формализация понятия алгоритма, оказались в точности такими же выразительными, как и машины Тьюринга:

- Нормальные алгорифмы Маркова
- Частично рекурсивные функции
- Машины Поста
- Машины Минского
- λ-исчисление
- Системы переписывания строк (полу-Туэ)
- RAM-машины
- ... ...

Такие модели, в конечном итоге описывающие возможности вычисления одного и того же класса функций, принято называть алгоритмически полными

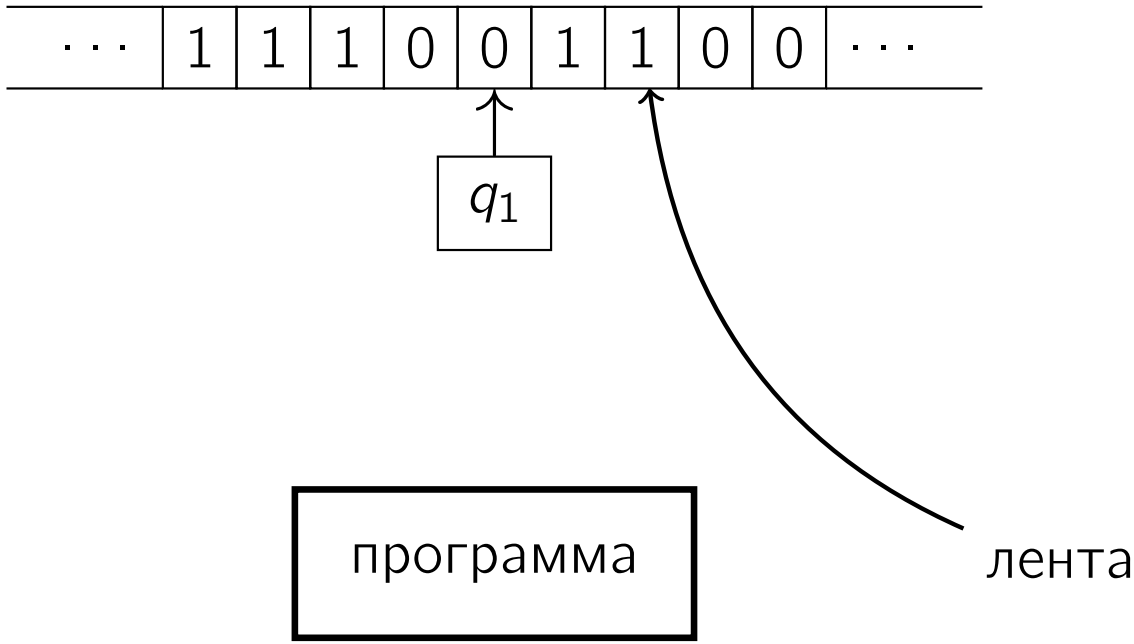
Вступление

А является ли модель ХЛП алгоритмически полной, или же есть что-то такое, что могут вычислять полные модели, а ХЛП не могут?

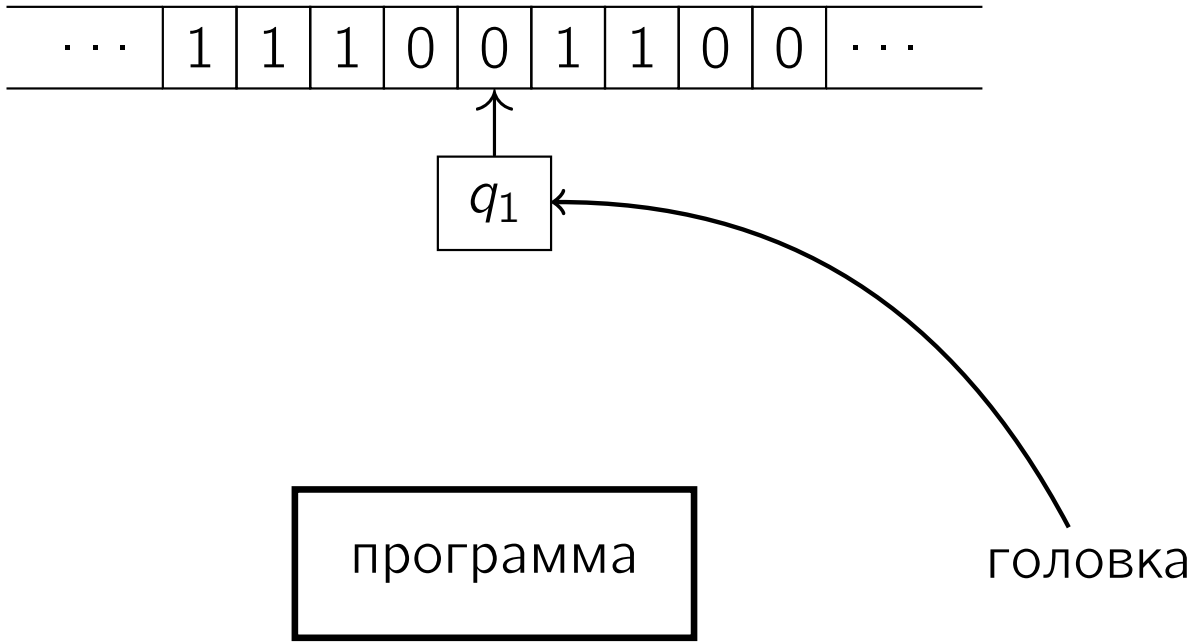
Обсуждение этого вопроса начнём с напоминания о том, что такое машина Тьюринга

Существует немало вариантов этой модели и немало способов её введения, поэтому дальнейшее можно расценивать не только как напоминание, но и как выбор варианта и системы обозначений

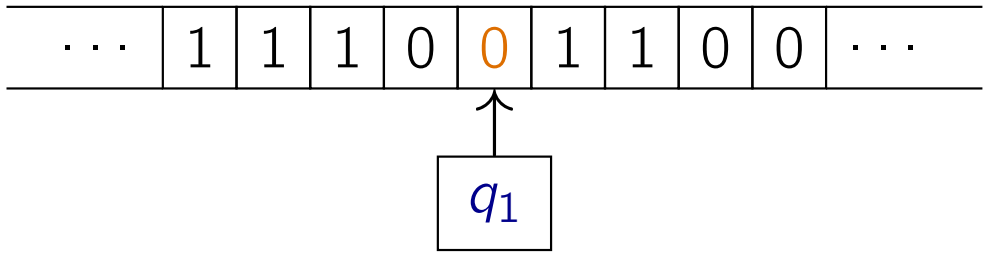
Машины Тьюринга (МТ)

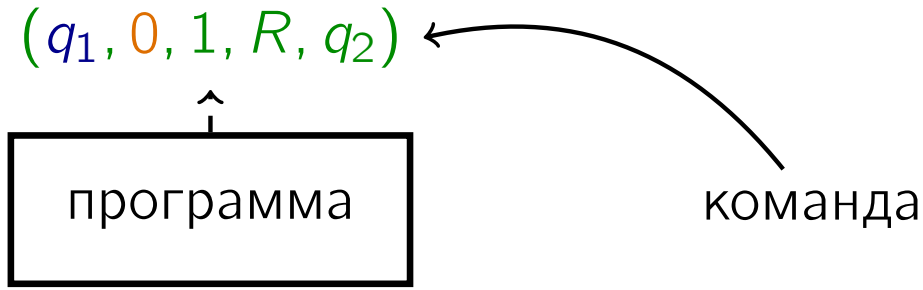


Машины Тьюринга (МТ)



Машины Тьюринга (МТ)

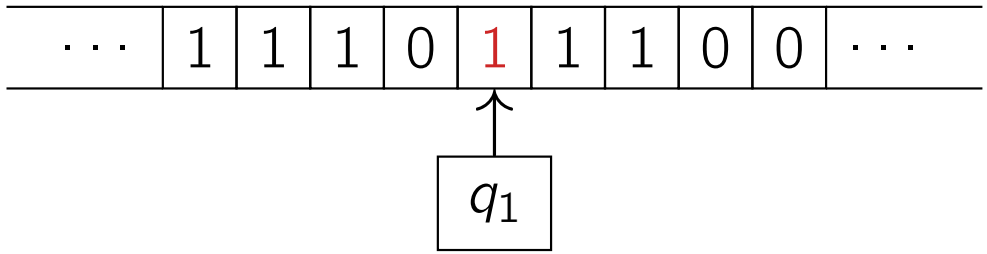




Шаг вычисления выглядит так:

- по текущему состоянию
  и обозреваемому символу выбираем команду

Машины Тьюринга (МТ)

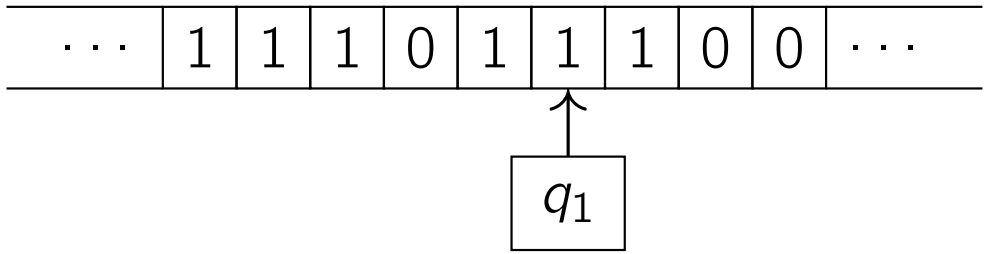


$$(q_1, 0, 1, R, q_2)$$

Шаг вычисления выглядит так:

- по текущему состоянию
  и обозреваемому символу выбираем команду
- записываем в ячейку новый символ

Машины Тьюринга (МТ)

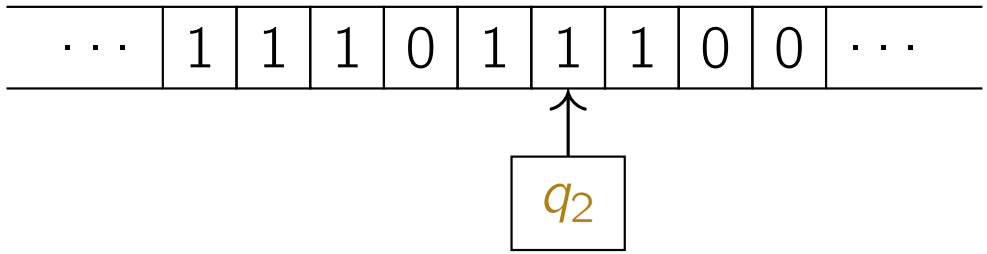


$(q_1, 0, 1, R, q_2)$

Шаг вычисления выглядит так:

- по текущему состоянию
  и обозреваемому символу выбираем команду
- записываем в ячейку новый символ
- сдвигаем головку

Машины Тьюринга (МТ)



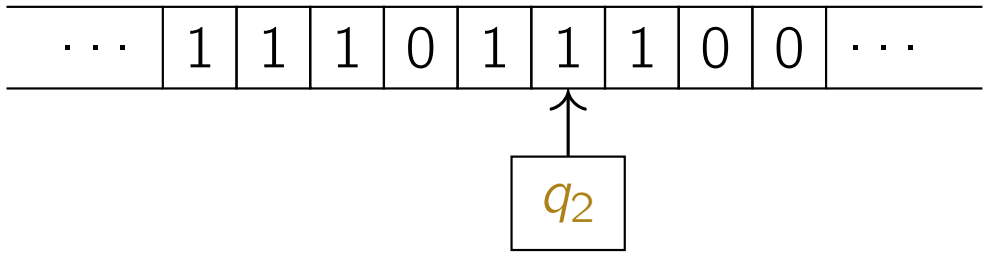
$(q_1, 0, 1, R, q_2)$

программа

Шаг вычисления выглядит так:

- по текущему состоянию
  и обозреваемому символу выбираем команду
- записываем в ячейку новый символ
- сдвигаем головку
- меняем состояние

Машины Тьюринга (МТ)



$$(q_1, 0, 1, R, q_2)$$

Шаг вычисления выглядит так:

- по текущему состоянию
  и обозреваемому символу выбираем команду
- записываем в ячейку новый символ
- сдвигаем головку
- меняем состояние

Определим всё это по порядку, детально и строго

МТ: синтаксис

Алфавит — это непустое конечное множество символов (букв)

Машина Тьюринга (МТ)[1] — это система $(\mathfrak{A}, \Lambda, \mathcal{Q}, q_0, q_f, \pi)$, где

- $\mathfrak{A}$ — алфавит ленты
- $\Lambda \in \mathfrak{A}$ — пустой символ
- $\mathcal{Q}$ — алфавит состояний
- $q_0, q_f \in \mathcal{Q}$ — соответственно начальное состояние и заключительное состояние
- $\pi : (\mathcal{Q} \setminus \{q_f\}) \times \mathfrak{A} \to \mathfrak{A} \times \{L, R\} \times \mathcal{Q}$ — программа

Программа МТ также будет пониматься как множество команд:

$$(q, a, b, S, p) \in \pi \Leftrightarrow \pi(q, a) = (b, S, p)$$

1 Здесь обсуждается только один из вариантов определения машины Тьюринга, удобный для сравнения с ХЛП

МТ: семантика

$$M = (\mathfrak{A}, \Lambda, \mathcal{Q}, q_0, q_f, \pi)$$

Слово в алфавите Σ — это конечная последовательность букв из Σ

В записи слов принято опускать
разделители (запятые) между символами:

$$\cancel{x_1, x_2, \ldots, x_n} \qquad x_1 x_2 \ldots x_n$$

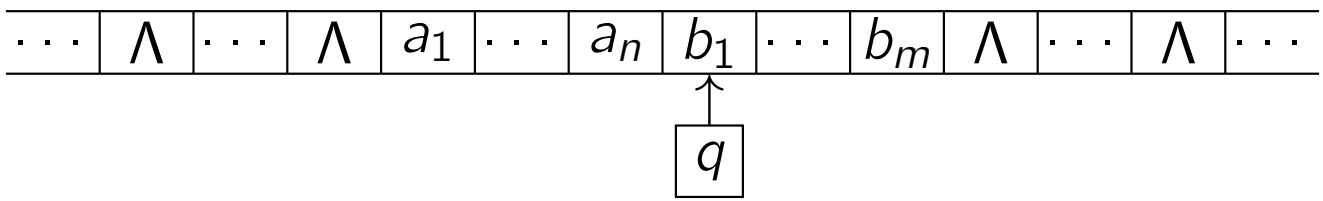
Σ^* — множество всех слов в алфавите Σ

Σ^+ — множество всех непустых слов в алфавите Σ

Ленточное слово — это слово в алфавите $\mathfrak{A}$

Конфигурация машины Тьюринга — это набор (α, q, β),
где $\alpha, \beta \in \mathfrak{A}^+$ и $q \in \mathcal{Q}$

Пояснение: конфигурация $(a_1 \ldots a_n, q, b_1 \ldots b_m)$ означает, что МТ находится в состоянии q, на ленте записано слово $a_1 \ldots a_n b_1 \ldots b_m$, окружённое пустыми символами, и обозревается символ b_1



МТ: семантика

$$M = (\mathfrak{A}, \Lambda, \mathcal{Q}, q_0, q_f, \pi)$$

Способ преобразования конфигураций командой C можно задать как двуместное отношение $\to_C$ на множестве конфигураций

Если $C = (q, a, b, R, p)$, то $\to_C$ состоит из следующих пар:

$$(\alpha, q, ax\beta) \to_C (\alpha b, p, x\beta)$$
$$(\alpha, q, a) \to_C (\alpha b, p, \Lambda)$$

Если $C = (q, a, b, L, p)$, то $\to_C$ состоит из следующих пар:

$$(\alpha xy, q, a\beta) \to_C (\alpha x, p, yb\beta)$$
$$(y, q, a\beta) \to_C (\Lambda, p, yb\beta)$$

$$(\alpha, \beta \in \mathfrak{A}^*;\ x, y \in \mathfrak{A})$$

Отношение переходов $\to_M$ МТ M — это объединение отношений $\to_C$ по всем командам C МТ M

Трасса МТ M — это последовательность конфигураций, в которой каждая пара соседних конфигураций σ_i, σ_{i+1} входит в отношение переходов: $\sigma_i \to_M \sigma_{i+1}$

$\to_M^*$ — это рефлексивно-транзитивное замыкание отношения $\to_M$:

$\sigma_1 \to_M^* \sigma_2 \quad \Leftrightarrow \quad$ существует трасса с началом σ_1 и концом σ_2

МТ: семантика

$$M = (\mathfrak{A}, \Lambda, \mathcal{Q}, q_0, q_f, \pi)$$

Конфигурация (α, q, β) — заключительная, если $q = q_f$

Вычисление МТ M на ленточном слове w — это трасса,

- начинающаяся в конфигурации $(\Lambda, q_0, w\Lambda)$ и
- либо бесконечная,
  либо оканчивающаяся в заключительной конфигурации

Утверждение. Для любой МТ M и любого ленточного слова w существует единственное вычисление M на w

Доказательство.
Достаточно заметить, что в соотношении $\sigma_i \to_M \sigma_{i+1}$

- σ_i может быть любой незаключительной конфигурацией
  и не может быть заключительной, и
- σ_{i+1} однозначно определяется конфигурацией σ_i ▼

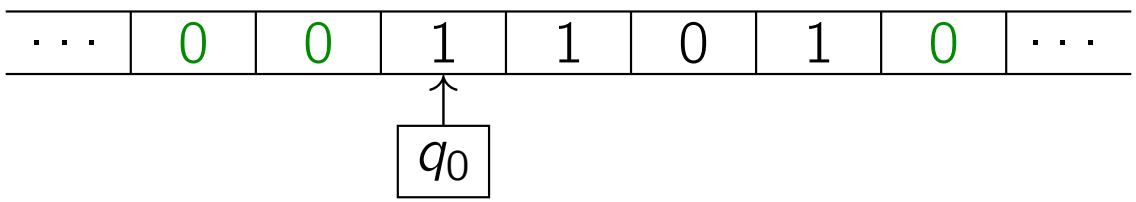
МТ: пример

Пример: $M = (\{0,1\}, 0, \{q_0, q_1, q_f\}, q_0, q_f, \pi)$, где:

$$\pi(q_0, 0) = (0, L, q_1) \qquad \pi(q_1, 0) = (0, R, q_f)$$
$$\pi(q_0, 1) = (1, R, q_0) \qquad \pi(q_1, 1) = (0, L, q_1)$$

Вычисление M на слове 1101:

$$(0, \quad q_0, \quad 11010)$$



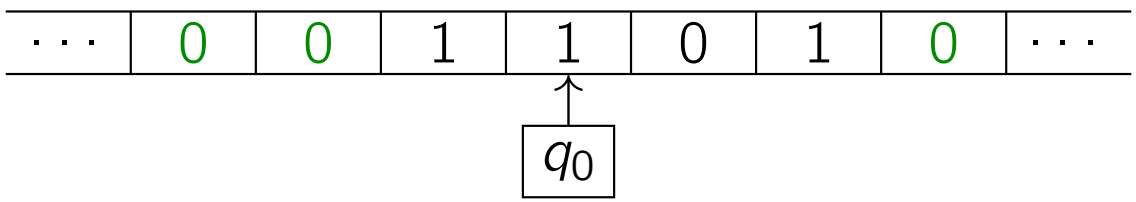
МТ: пример

Пример: $M = (\{0,1\}, 0, \{q_0, q_1, q_f\}, q_0, q_f, \pi)$, где:

$$\pi(q_0, 0) = (0, L, q_1) \qquad \pi(q_1, 0) = (0, R, q_f)$$
$$\pi(q_0, 1) = (1, R, q_0) \qquad \pi(q_1, 1) = (0, L, q_1)$$

Вычисление M на слове 1101:

$$\begin{array}{lll} (0, & q_0, & 11010) \\ (01, & q_0, & 1010) \end{array} \quad \rightarrow_M$$



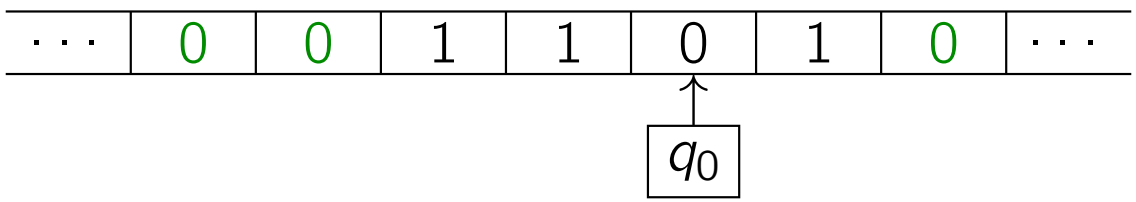
МТ: пример

Пример: $M = (\{0,1\}, 0, \{q_0, q_1, q_f\}, q_0, q_f, \pi)$, где:

$$\pi(q_0, 0) = (0, L, q_1) \qquad \pi(q_1, 0) = (0, R, q_f)$$
$$\pi(q_0, 1) = (1, R, q_0) \qquad \pi(q_1, 1) = (0, L, q_1)$$

Вычисление M на слове 1101:

$$\begin{aligned} (0, &\ q_0, &\ 11010) &\ \to_M \\ (01, &\ q_0, &\ 1010) &\ \to_M \\ (011, &\ q_0, &\ 010) & \end{aligned}$$



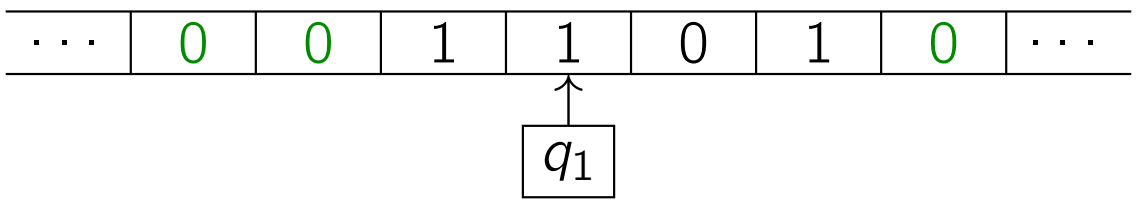
МТ: пример

Пример: $M = (\{0,1\}, 0, \{q_0, q_1, q_f\}, q_0, q_f, \pi)$, где:

$$\pi(q_0, 0) = (0, L, q_1) \qquad \pi(q_1, 0) = (0, R, q_f)$$
$$\pi(q_0, 1) = (1, R, q_0) \qquad \pi(q_1, 1) = (0, L, q_1)$$

Вычисление M на слове 1101:

$$\begin{aligned} (0, &\ q_0, &\ 11010) &\ \to_M \\ (01, &\ q_0, &\ 1010) &\ \to_M \\ (011, &\ q_0, &\ 010) &\ \to_M \\ (01, &\ q_1, &\ 1010) & \end{aligned}$$



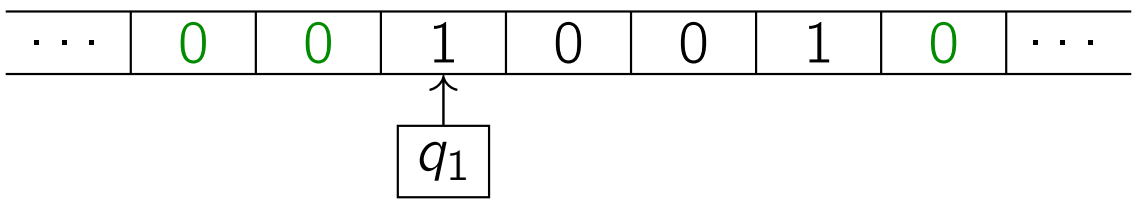
МТ: пример

Пример: $M = (\{0,1\}, 0, \{q_0, q_1, q_f\}, q_0, q_f, \pi)$, где:

$$\pi(q_0, 0) = (0, L, q_1) \qquad \pi(q_1, 0) = (0, R, q_f)$$
$$\pi(q_0, 1) = (1, R, q_0) \qquad \pi(q_1, 1) = (0, L, q_1)$$

Вычисление M на слове 1101:

$$\begin{aligned}
(0, &\ q_0, &\ 11010) &\ \to_M \\
(01, &\ q_0, &\ 1010) &\ \to_M \\
(011, &\ q_0, &\ 010) &\ \to_M \\
(01, &\ q_1, &\ 1010) &\ \to_M \\
(0, &\ q_1, &\ 10010)
\end{aligned}$$



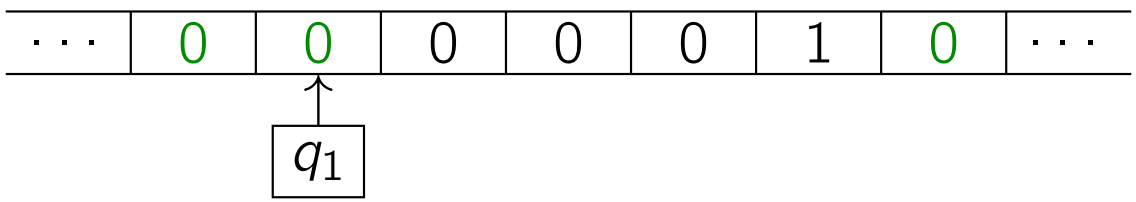
МТ: пример

Пример: $M = (\{0,1\}, 0, \{q_0, q_1, q_f\}, q_0, q_f, \pi)$, где:

$$\pi(q_0, 0) = (0, L, q_1) \qquad \pi(q_1, 0) = (0, R, q_f)$$
$$\pi(q_0, 1) = (1, R, q_0) \qquad \pi(q_1, 1) = (0, L, q_1)$$

Вычисление M на слове 1101:

$$\begin{aligned}
(0, &\ q_0, &\ 11010) &\ \to_M \\
(01, &\ q_0, &\ 1010) &\ \to_M \\
(011, &\ q_0, &\ 010) &\ \to_M \\
(01, &\ q_1, &\ 1010) &\ \to_M \\
(0, &\ q_1, &\ 10010) &\ \to_M \\
(0, &\ q_1, &\ 000010)
\end{aligned}$$



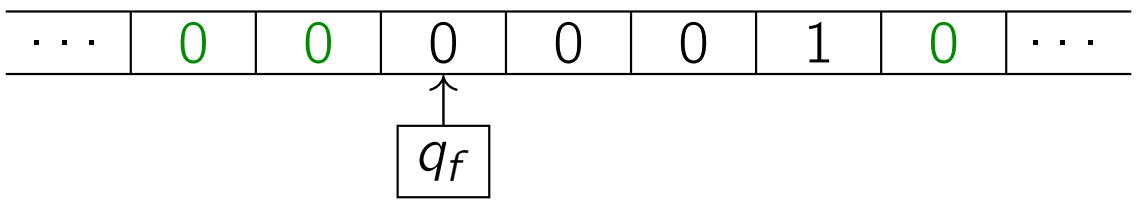
МТ: пример

Пример: $M = (\{0,1\}, 0, \{q_0, q_1, q_f\}, q_0, q_f, \pi)$, где:

$$\pi(q_0, 0) = (0, L, q_1) \qquad \pi(q_1, 0) = (0, R, q_f)$$
$$\pi(q_0, 1) = (1, R, q_0) \qquad \pi(q_1, 1) = (0, L, q_1)$$

Вычисление M на слове 1101:

$$\begin{aligned}
(0, &\ q_0, &\ 11010) &\ \to_M \\
(01, &\ q_0, &\ 1010) &\ \to_M \\
(011, &\ q_0, &\ 010) &\ \to_M \\
(01, &\ q_1, &\ 1010) &\ \to_M \\
(0, &\ q_1, &\ 10010) &\ \to_M \\
(0, &\ q_1, &\ 000010) &\ \to_M \\
(00, &\ q_f, &\ 00010) &
\end{aligned}$$



Математическая логика и логическое программирование

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

Блок 40

Моделирование машин Тьюринга
хорновскими логическими программами

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

ВМК МГУ, 2023/2024, осенний семестр

Чтобы показать, насколько широки вычислительные возможности ХЛП, опишем способ преобразования МТ $M = (\mathcal{A}, \Lambda, \mathcal{Q}, q_0, q_f, \pi)$ в ХЛП $\mathcal{P}_M$, особым образом воспроизводящую вычисления M

Символы из $\mathcal{A} \cup \mathcal{Q}$ будем использовать в ХЛП в качестве констант

Ленточному слову $w = \mathbf{a}_1 \ldots \mathbf{a}_k$ сопоста́вим список $\tau_w = \mathbf{a}_1\mathbf{.} \ldots \mathbf{.a}_k\mathbf{.nil}$

w^- — так будем записывать зеркальный образ слова w: если $w = \mathbf{a}_1 \ldots \mathbf{a}_k$, то $w^- = \mathbf{a}_k \ldots \mathbf{a}_1$

Конфигурации $\sigma = (\alpha, \mathbf{q}, \beta)$ МТ сопоста́вим список из трёх элементов $\tau_\sigma = (\tau_{\alpha^-})\mathbf{.q.}(\tau_\beta)\mathbf{.nil}$

Например, если $\sigma = (\mathbf{001}, \mathbf{q}, \mathbf{110})$, то $\tau_\sigma = (\mathbf{1.0.0.nil})\mathbf{.q.}(\mathbf{1.1.0.nil})\mathbf{.nil}$

В программе будем использовать только один предикатный символ p, вкладывая в него такой смысл: $\mathrm{p}(\tau_\sigma) =$ «σ — текущая конфигурация МТ M»

Каждой команде C МТ сопоста́вим два правила ХЛП $\mathcal{R}_C$, $\mathcal{S}_C$:

- Если $C = (\mathbf{q}, \mathbf{a}, \mathbf{b}, R, \mathbf{p})$, то

$$\begin{aligned} \mathcal{R}_C = \quad & \mathrm{p}(\mathtt{A}.\mathbf{q}.(\mathbf{a}.\mathtt{X}.\mathtt{B}).\mathbf{nil}) \quad \leftarrow \mathrm{p}((\mathbf{b}.\mathtt{A}).\mathbf{p}.(\mathtt{X}.\mathtt{B}).\mathbf{nil}); \\ \mathcal{S}_C = \quad & \mathrm{p}(\mathtt{A}.\mathbf{q}.(\mathbf{a}.\mathbf{nil}).\mathbf{nil}) \quad \leftarrow \mathrm{p}((\mathbf{b}.\mathtt{A}).\mathbf{p}.(\mathbf{\Lambda}.\mathbf{nil}).\mathbf{nil}); \end{aligned}$$

  Напоминание:

$$\begin{aligned} (\alpha, \mathbf{q}, \mathbf{a}x\beta) \quad & \to_C \quad (\alpha\mathbf{b}, \mathbf{p}, x\beta) \\ (\alpha, \mathbf{q}, \mathbf{a}) \quad & \to_C \quad (\alpha\mathbf{b}, \mathbf{p}, \mathbf{\Lambda}) \end{aligned}$$

- Если $C = (\mathbf{q}, \mathbf{a}, \mathbf{b}, L, \mathbf{p})$, то

$$\begin{aligned} \mathcal{R}_C = \quad & \mathrm{p}((\mathtt{Y}.\mathtt{X}.\mathtt{A}).\mathbf{q}.(\mathbf{a}.\mathtt{B}).\mathbf{nil}) \quad \leftarrow \mathrm{p}((\mathtt{X}.\mathtt{A}).\mathbf{p}.(\mathtt{Y}.\mathbf{b}.\mathtt{B}).\mathbf{nil}); \\ \mathcal{S}_C = \quad & \mathrm{p}((\mathtt{Y}.\mathbf{nil}).\mathbf{q}.(\mathbf{a}.\mathtt{B}).\mathbf{nil}) \quad \leftarrow \mathrm{p}((\mathbf{\Lambda}.\mathbf{nil}).\mathbf{p}.(\mathtt{Y}.\mathbf{b}.\mathtt{B}).\mathbf{nil}); \end{aligned}$$

  Напоминание:

$$\begin{aligned} (\alpha xy, \mathbf{q}, \mathbf{a}\beta) \quad & \to_C \quad (\alpha x, \mathbf{p}, y\mathbf{b}\beta) \\ (y, \mathbf{q}, \mathbf{a}\beta) \quad & \to_C \quad (\mathbf{\Lambda}, \mathbf{p}, y\mathbf{b}\beta) \end{aligned}$$

Лемма 1. **Для любых конфигурации σ, запроса $\mathcal{Q}$ и команды C верно:**

$?\mathrm{p}(\tau_\sigma) \xrightarrow{\mathcal{R}_C} \mathcal{Q}$ **или** $?\mathrm{p}(\tau_\sigma) \xrightarrow{\mathcal{S}_C} \mathcal{Q} \Leftrightarrow$

существует конфигурация σ', такая что $\mathcal{Q} = ?\mathrm{p}(\tau_{\sigma'})$ и $\sigma \to_C \sigma'$

Доказательство. Очевидно? (С учётом напоминаний)

Машине Тьюринга M с программой π сопоста́вим ХЛП $\mathcal{P}_M$, состоящую из правил $\mathcal{R}_C$ и $\mathcal{S}_C$ для всех $C \in \pi$ в любом порядке

Например, если $M = (\{\mathbf{0}, \mathbf{1}\}, \mathbf{0}, \{\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_f\}, \mathbf{q}_0, \mathbf{q}_f, \pi)$, где

$$\pi(\mathbf{q}_0, \mathbf{0}) = (\mathbf{0}, L, \mathbf{q}_1), \qquad \pi(\mathbf{q}_1, \mathbf{0}) = (\mathbf{0}, R, \mathbf{q}_f),$$
$$\pi(\mathbf{q}_0, \mathbf{1}) = (\mathbf{1}, R, \mathbf{q}_0), \qquad \pi(\mathbf{q}_1, \mathbf{1}) = (\mathbf{0}, L, \mathbf{q}_1),$$

то программа $\mathcal{P}_M$ содержит 8 правил:

$$\mathrm{p}((\mathtt{Y}.\mathtt{X}.\mathtt{A}).\mathbf{q}_0.(\mathbf{0}.\mathtt{B}).\mathbf{nil}) \leftarrow \mathrm{p}((\mathtt{X}.\mathtt{A}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil});$$
$$\mathrm{p}((\mathtt{Y}.\mathbf{nil}).\mathbf{q}_0.(\mathbf{0}.\mathtt{B}).\mathbf{nil}) \leftarrow \mathrm{p}((\mathbf{\Lambda}.\mathbf{nil}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil});$$

$$\mathrm{p}(\mathtt{A}.\mathbf{q}_0.(\mathbf{1}.\mathtt{X}.\mathtt{B}).\mathbf{nil}) \leftarrow \mathrm{p}((\mathbf{1}.\mathtt{A}).\mathbf{q}_0.(\mathtt{X}.\mathtt{B}).\mathbf{nil});$$
$$\mathrm{p}(\mathtt{A}.\mathbf{q}_0.(\mathbf{1}.\mathbf{nil}).\mathbf{nil}) \leftarrow \mathrm{p}((\mathbf{1}.\mathtt{A}).\mathbf{q}_0.(\mathbf{\Lambda}.\mathbf{nil}).\mathbf{nil});$$

$$\mathrm{p}(\mathtt{A}.\mathbf{q}_1.(\mathbf{0}.\mathtt{X}.\mathtt{B}).\mathbf{nil}) \leftarrow \mathrm{p}((\mathbf{0}.\mathtt{A}).\mathbf{q}_f.(\mathtt{X}.\mathtt{B}).\mathbf{nil});$$
$$\mathrm{p}(\mathtt{A}.\mathbf{q}_1.(\mathbf{0}.\mathbf{nil}).\mathbf{nil}) \leftarrow \mathrm{p}((\mathbf{0}.\mathtt{A}).\mathbf{q}_f.(\mathbf{\Lambda}.\mathbf{nil}).\mathbf{nil});$$

$$\mathrm{p}((\mathtt{Y}.\mathtt{X}.\mathtt{A}).\mathbf{q}_1.(\mathbf{1}.\mathtt{B}).\mathbf{nil}) \leftarrow \mathrm{p}((\mathtt{X}.\mathtt{A}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil});$$
$$\mathrm{p}((\mathtt{Y}.\mathbf{nil}).\mathbf{q}_1.(\mathbf{1}.\mathtt{B}).\mathbf{nil}) \leftarrow \mathrm{p}((\mathbf{\Lambda}.\mathbf{nil}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil});$$

$\mathcal{Q}_1 \to_{\mathcal{P}} \mathcal{Q}_2$ — так будем обозначать тот факт, что хотя бы для одного правила $\mathcal{R}$ ХЛП $\mathcal{P}$ верно $\mathcal{Q}_1 \xrightarrow{\mathcal{R}} \mathcal{Q}_2$

Лемма 2. Для любых МТ M, её конфигурации σ и запроса $\mathcal{Q}$ верно:
$?\mathrm{p}(\tau_\sigma) \to_{\mathcal{P}_M} \mathcal{Q} \Leftrightarrow$
существует конфигурация σ', такая что $\mathcal{Q} = ?\mathrm{p}(\sigma')$ и $\sigma \to_M \sigma'$

Доказательство

$?\mathrm{p}(\tau_\sigma) \to_{\mathcal{P}_M} \mathcal{Q}$

$\Leftrightarrow$ (по устройству $\mathcal{P}_M$)

$\exists$ команда C МТ M, такая что $?\mathrm{p}(\tau_\sigma) \to_{\mathcal{R}_C} \mathcal{Q}$ или $?\mathrm{p}(\tau_\sigma) \to_{\mathcal{S}_C} \mathcal{Q}$

$\Leftrightarrow$ (по лемме 1)

$\exists$ команда C МТ M и конфигурация σ', такая что $\mathcal{Q} = ?\mathrm{p}(\tau_{\sigma'})$ и $\sigma \to_C \sigma'$

$\Leftrightarrow$ (по определению $\to_M$)

$\exists$ конфигурация σ', такая что $\mathcal{Q} = ?\mathrm{p}(\tau_{\sigma'})$ и $\sigma \to_M \sigma'$ ▼

Теорема (о моделировании машин Тьюринга хорновскими ЛП).
Для любых МТ M и конфигурации σ последовательность запросов

$$?\mathrm{p}(\tau_\sigma),\ ?\mathrm{p}(t_1),\ ?\mathrm{p}(t_2), \ldots$$

является вычислением ХЛП $\mathcal{P}_M$ тогда и только тогда, когда существуют конфигурации $\sigma_1, \sigma_2, \ldots$, такие что

- $t_1 = \tau_{\sigma_1},\ t_2 = \tau_{\sigma_2}, \ldots$ **и**
- $\sigma \to_M \sigma_1 \to_M \sigma_2 \to_M \ldots$

Доказательство. Следует из леммы 2

Вычисление ХЛП, порождённое запросом $\mathcal{Q}$, будем для краткости называть вычислением на запросе $\mathcal{Q}$

Вычисление ХЛП назовём непродолжаемым, если оно успешное, тупиковое или бесконечное

Следствие. Для любой МТ M и любой её конфигурации σ существует ровно одно непродолжаемое вычисление программы $\mathcal{P}_M$ на $?\mathrm{p}(\tau_\sigma)$

Пример: $M = (\{\mathbf{0}, \mathbf{1}\}, \mathbf{0}, \{\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_f\}, \mathbf{q}_0, \mathbf{q}_f, \pi)$, где

$$\pi(\mathbf{q}_0, \mathbf{0}) = (\mathbf{0}, L, \mathbf{q}_1), \qquad \pi(\mathbf{q}_1, \mathbf{0}) = (\mathbf{0}, R, \mathbf{q}_f),$$
$$\pi(\mathbf{q}_0, \mathbf{1}) = (\mathbf{1}, R, \mathbf{q}_0), \qquad \pi(\mathbf{q}_1, \mathbf{1}) = (\mathbf{0}, L, \mathbf{q}_1),$$

$\mathcal{P}_M$:

$$\begin{array}{lll}
1: & \mathrm{p}((\mathtt{Y}.\mathtt{X}.\mathtt{A}).\mathbf{q}_0.(\mathbf{0}.\mathtt{B}).\mathbf{nil}) & \leftarrow \mathrm{p}((\mathtt{X}.\mathtt{A}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil});\\
2: & \mathrm{p}((\mathtt{Y}.\mathbf{nil}).\mathbf{q}_0.(\mathbf{0}.\mathtt{B}).\mathbf{nil}) & \leftarrow \mathrm{p}((\mathbf{\Lambda}.\mathbf{nil}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil});\\
3: & \mathrm{p}(\mathtt{A}.\mathbf{q}_0.(\mathbf{1}.\mathtt{X}.\mathtt{B}).\mathbf{nil}) & \leftarrow \mathrm{p}((\mathbf{1}.\mathtt{A}).\mathbf{q}_0.(\mathtt{X}.\mathtt{B}).\mathbf{nil});\\
4: & \mathrm{p}(\mathtt{A}.\mathbf{q}_0.(\mathbf{1}.\mathbf{nil}).\mathbf{nil}) & \leftarrow \mathrm{p}((\mathbf{1}.\mathtt{A}).\mathbf{q}_0.(\mathbf{\Lambda}.\mathbf{nil}).\mathbf{nil});\\
5: & \mathrm{p}(\mathtt{A}.\mathbf{q}_1.(\mathbf{0}.\mathtt{X}.\mathtt{B}).\mathbf{nil}) & \leftarrow \mathrm{p}((\mathbf{0}.\mathtt{A}).\mathbf{q}_f.(\mathtt{X}.\mathtt{B}).\mathbf{nil});\\
6: & \mathrm{p}(\mathtt{A}.\mathbf{q}_1.(\mathbf{0}.\mathbf{nil}).\mathbf{nil}) & \leftarrow \mathrm{p}((\mathbf{0}.\mathtt{A}).\mathbf{q}_f.(\mathbf{\Lambda}.\mathbf{nil}).\mathbf{nil});\\
7: & \mathrm{p}((\mathtt{Y}.\mathtt{X}.\mathtt{A}).\mathbf{q}_1.(\mathbf{1}.\mathtt{B}).\mathbf{nil}) & \leftarrow \mathrm{p}((\mathtt{X}.\mathtt{A}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil});\\
8: & \mathrm{p}((\mathtt{Y}.\mathbf{nil}).\mathbf{q}_1.(\mathbf{1}.\mathtt{B}).\mathbf{nil}) & \leftarrow \mathrm{p}((\mathbf{\Lambda}.\mathbf{nil}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil});
\end{array}$$

Вычисление M на слове 1:

$$(\mathbf{0}, \mathbf{q}_0, \mathbf{10}) \to_M (\mathbf{01}, \mathbf{q}_0, \mathbf{0}) \to_M (\mathbf{0}, \mathbf{q}_1, \mathbf{10}) \to_M (\mathbf{0}, \mathbf{q}_1, \mathbf{000}) \to_M (\mathbf{00}, \mathbf{q}_f, \mathbf{00})$$

Соответствующее непродолжаемое вычисление $\mathcal{P}_M$:

$$?\mathrm{p}((\mathbf{0.nil}).\mathbf{q}_0.(\mathbf{1.0.nil}).\mathbf{nil}) \xrightarrow{3} ?\mathrm{p}((\mathbf{1.0.nil}).\mathbf{q}_0.(\mathbf{0.nil}).\mathbf{nil}) \xrightarrow{1}$$
$$?\mathrm{p}((\mathbf{0.nil}).\mathbf{q}_1.(\mathbf{1.0.nil}).\mathbf{nil}) \xrightarrow{8} ?\mathrm{p}((\mathbf{0.nil}).\mathbf{q}_1.(\mathbf{0.0.0.nil}).\mathbf{nil}) \xrightarrow{5}$$
$$?\mathrm{p}((\mathbf{0.0.nil}).\mathbf{q}_f.(\mathbf{0.0.nil}).\mathbf{nil})$$

$\mathcal{P}_M^1$ так обозначим ХЛП, получающуюся из $\mathcal{P}_M$ добавлением факта $\mathrm{p}(\mathtt{X}.\mathbf{q}_f.\mathtt{Y}.\mathbf{nil})$;, где $\mathbf{q}_f$ — заключительное состояние МТ M

$\mathcal{Q}_1 \to_{\mathcal{P}}^* \mathcal{Q}_2$ — так обозначим тот факт, что существует вычисление $\mathcal{P}$ на $\mathcal{Q}_1$, оканчивающееся запросом $\mathcal{Q}_2$ (то есть $\to_{\mathcal{P}}^*$ — рефлексивно-транзитивное замыкание отношения $\to_{\mathcal{P}}$)

Следствие. **Для любой МТ** $M = (\mathcal{A}, \mathbf{\Lambda}, \mathcal{Q}, \mathbf{q}_0, \mathbf{q}_f, \pi)$ **и любого ленточного сло́ва** w **верно следующее:**
вычисление M **на** w **конечно** $\Leftrightarrow ?\mathrm{p}(\tau_{(\Lambda,\mathbf{q}_0,w\Lambda)}) \to_{\mathcal{P}^1}^* \square$

Доказательство.

Вычисление M на w конечно

$\Leftrightarrow$ (по последней теореме)

$\exists$ ленточные слова w_1, w_2, такие что $?\mathrm{p}(\tau_{(\Lambda,\mathbf{q}_0,w\Lambda)}) \to_{\mathcal{P}_M}^* ?\mathrm{p}(\tau_{(w_1,\mathbf{q}_f,w_2)})$

$\Leftrightarrow$ (т.к. $\mathrm{p}(\tau_{(u,q,v)})$ унифицируемо с $p(\mathtt{X}.\mathbf{q}_f.\mathtt{Y}.\mathbf{nil}) \Leftrightarrow q = \mathbf{q}_f$)

$\exists$ ленточные слова w_1, w_2, такие что
$?\mathrm{p}(\tau_{(\Lambda,\mathbf{q}_0,w\Lambda)}) \to_{\mathcal{P}_M^1}^* ?\mathrm{p}(\tau_{(w_1,\mathbf{q}_f,w_2)}) \to_{\mathcal{P}_M^1} \square$ ▼

$\mathcal{P}^2_M$ — так для МТ M с программой π и заключительным состоянием $\mathbf{q}_f$ обозна́чим ХЛП, состоящую из следующих правил $\mathcal{R}^2_C$ и $\mathcal{S}^2_C$ для $C \in \pi$ и одного факта $\mathcal{F}$ в любом порядке:

- Если $C = (\mathbf{q}, \mathbf{a}, \mathbf{b}, R, \mathbf{p})$, то
$$\mathcal{R}^2_C = \mathrm{p}((\mathbf{b}.\mathtt{A}).\mathbf{p}.(\mathtt{X}.\mathtt{B}).\mathbf{nil}, \mathtt{Z}) \leftarrow \mathrm{p}(\mathtt{A}.\mathbf{q}.(\mathbf{a}.\mathtt{X}.\mathtt{B}).\mathbf{nil}, \mathtt{Z});$$
$$\mathcal{S}^2_C = \mathrm{p}((\mathbf{b}.\mathtt{A}).\mathbf{p}.(\mathbf{\Lambda}.\mathbf{nil}).\mathbf{nil}, \mathtt{Z}) \leftarrow \mathrm{p}(\mathtt{A}.\mathbf{q}.(\mathbf{a}.\mathbf{nil}).\mathbf{nil}, \mathtt{Z});$$
- Если $C = (\mathbf{q}, \mathbf{a}, \mathbf{b}, L, \mathbf{p})$, то
$$\mathcal{R}^2_C = \mathrm{p}((\mathtt{X}.\mathtt{A}).\mathbf{p}.(\mathtt{Y}.\mathbf{b}.\mathtt{B}).\mathbf{nil}, \mathtt{Z}) \leftarrow \mathrm{p}((\mathtt{Y}.\mathtt{X}.\mathtt{A}).\mathbf{q}.(\mathbf{a}.\mathtt{B}).\mathbf{nil}, \mathtt{Z});$$
$$\mathcal{S}^2_C = \mathrm{p}((\mathbf{\Lambda}.\mathbf{nil}).\mathbf{p}.(\mathtt{Y}.\mathbf{b}.\mathtt{B}).\mathbf{nil}, \mathtt{Z}) \leftarrow \mathrm{p}((\mathtt{Y}.\mathbf{nil}).\mathbf{q}.(\mathbf{a}.\mathtt{B}).\mathbf{nil}, \mathtt{Z});$$
- $\mathcal{F} = \mathrm{p}(\mathtt{X}.\mathbf{q}_f.\mathtt{Y}.\mathbf{nil}, \mathtt{X}.\mathbf{q}_f.\mathtt{Y}.\mathbf{nil})$;

$\mathcal{P}^2_M$ отличается от $\mathcal{P}^1_M$ только вторым аргументом p, изменяющимся только при применении $\mathcal{F}$ унификацией с первым аргументом (списком, представляющим заключительную конфигурацию M)

Следствие. **Вычисление МТ $M = (\mathcal{A}, \mathbf{\Lambda}, \mathcal{Q}, \mathbf{q}_0, \mathbf{q}_f, \pi)$ на w конечно и оканчивается конфигурацией σ $\Leftrightarrow$ программа $\mathcal{P}^2_M$ имеет единственное непродолжаемое вычисление на $?\mathrm{p}(\tau_{(\mathbf{\Lambda}, \mathbf{q}_0, w\mathbf{\Lambda})}, \mathtt{X})$, и это успешное вычисление с результатом $\{\mathtt{X}/\tau_\sigma\}$**

Пример: $M = (\{\mathbf{0}, \mathbf{1}\}, \mathbf{0}, \{\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_f\}, \mathbf{q}_0, \mathbf{q}_f, \pi)$, где

$$\pi(\mathbf{q}_0, \mathbf{0}) = (\mathbf{0}, L, \mathbf{q}_1), \qquad \pi(\mathbf{q}_1, \mathbf{0}) = (\mathbf{0}, R, \mathbf{q}_f),$$
$$\pi(\mathbf{q}_0, \mathbf{1}) = (\mathbf{1}, R, \mathbf{q}_0), \qquad \pi(\mathbf{q}_1, \mathbf{1}) = (\mathbf{0}, L, \mathbf{q}_1),$$

$\mathcal{P}_M^2$:

$$\begin{aligned}
1: &\ \mathrm{p}((\mathtt{Y}.\mathtt{X}.\mathtt{A}).\mathbf{q}_0.(\mathbf{0}.\mathtt{B}).\mathbf{nil}, \mathtt{Z}) &&\leftarrow \mathrm{p}((\mathtt{X}.\mathtt{A}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil}, \mathtt{Z});\\
2: &\ \mathrm{p}((\mathtt{Y}.\mathbf{nil}).\mathbf{q}_0.(\mathbf{0}.\mathtt{B}).\mathbf{nil}, \mathtt{Z}) &&\leftarrow \mathrm{p}((\mathbf{\Lambda}.\mathbf{nil}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil}, \mathtt{Z});\\
3: &\ \mathrm{p}(\mathtt{A}.\mathbf{q}_0.(\mathbf{1}.\mathtt{X}.\mathtt{B}).\mathbf{nil}, \mathtt{Z}) &&\leftarrow \mathrm{p}((\mathbf{1}.\mathtt{A}).\mathbf{q}_0.(\mathtt{X}.\mathtt{B}).\mathbf{nil}, \mathtt{Z});\\
4: &\ \mathrm{p}(\mathtt{A}.\mathbf{q}_0.(\mathbf{1}.\mathbf{nil}).\mathbf{nil}, \mathtt{Z}) &&\leftarrow \mathrm{p}((\mathbf{1}.\mathtt{A}).\mathbf{q}_0.(\mathbf{\Lambda}.\mathbf{nil}).\mathbf{nil}, \mathtt{Z});\\
5: &\ \mathrm{p}(\mathtt{A}.\mathbf{q}_1.(\mathbf{0}.\mathtt{X}.\mathtt{B}).\mathbf{nil}, \mathtt{Z}) &&\leftarrow \mathrm{p}((\mathbf{0}.\mathtt{A}).\mathbf{q}_f.(\mathtt{X}.\mathtt{B}).\mathbf{nil}, \mathtt{Z});\\
6: &\ \mathrm{p}(\mathtt{A}.\mathbf{q}_1.(\mathbf{0}.\mathbf{nil}).\mathbf{nil}, \mathtt{Z}) &&\leftarrow \mathrm{p}((\mathbf{0}.\mathtt{A}).\mathbf{q}_f.(\mathbf{\Lambda}.\mathbf{nil}).\mathbf{nil}, \mathtt{Z});\\
7: &\ \mathrm{p}((\mathtt{Y}.\mathtt{X}.\mathtt{A}).\mathbf{q}_1.(\mathbf{1}.\mathtt{B}).\mathbf{nil}, \mathtt{Z}) &&\leftarrow \mathrm{p}((\mathtt{X}.\mathtt{A}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil}, \mathtt{Z});\\
8: &\ \mathrm{p}((\mathtt{Y}.\mathbf{nil}).\mathbf{q}_1.(\mathbf{1}.\mathtt{B}).\mathbf{nil}, \mathtt{Z}) &&\leftarrow \mathrm{p}((\mathbf{\Lambda}.\mathbf{nil}).\mathbf{q}_1.(\mathtt{Y}.\mathbf{0}.\mathtt{B}).\mathbf{nil}, \mathtt{Z});\\
9: &\ \mathrm{p}(\mathtt{X}.\mathbf{q}_f.\mathtt{Y}.\mathbf{nil}, \mathtt{X}.\mathbf{q}_f.\mathtt{Y}.\mathbf{nil})
\end{aligned}$$

Вычисление M на слове 1:

$$(\mathbf{0}, \mathbf{q}_0, \mathbf{10}) \to_M (\mathbf{01}, \mathbf{q}_0, \mathbf{0}) \to_M (\mathbf{0}, \mathbf{q}_1, \mathbf{10}) \to_M (\mathbf{0}, \mathbf{q}_1, \mathbf{000}) \to_M (\mathbf{00}, \mathbf{q}_f, \mathbf{00})$$

Соответствующее непродолжаемое вычисление $\mathcal{P}_M$:

$$?\mathrm{p}((\mathbf{0.nil}).\mathbf{q}_0.(\mathbf{1.0.nil}).\mathbf{nil}, \mathtt{X}) \xrightarrow{3} ?\mathrm{p}((\mathbf{1.0.nil}).\mathbf{q}_0.(\mathbf{0.nil}).\mathbf{nil}, \mathtt{X}) \xrightarrow{1}$$
$$?\mathrm{p}((\mathbf{0.nil}).\mathbf{q}_1.(\mathbf{1.0.nil}).\mathbf{nil}, \mathtt{X}) \xrightarrow{8} ?\mathrm{p}((\mathbf{0.nil}).\mathbf{q}_1.(\mathbf{0.0.0.nil}).\mathbf{nil}, \mathtt{X}) \xrightarrow{5}$$
$$?\mathrm{p}((\mathbf{0.0.nil}).\mathbf{q}_f.(\mathbf{0.0.nil}).\mathbf{nil}, \mathtt{X}) \xrightarrow{9,\{\mathtt{X}/(\mathbf{0.0.nil}).\mathbf{q}_f.(\mathbf{0.0.nil}).\mathbf{nil},\dots\}} \square$$

Таким образом, ХЛП умеют

- пошагово воспроизводить вычисления произвольных МТ и
- вычислять всё то, что умеют вычислять МТ

То есть ХЛП являются алгоритмически полными: с их помощью можно решить любую задачу, имеющую хоть какое-нибудь алгоритмическое решение

Оборотная сторона такой выразительности языка — это то, что анализ поведения ХЛП настолько же труден, насколько и для МТ

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, 2023/2024, 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 бóльшая 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 $\mathfrak{T}$ можно понимать как отображение $\mathfrak{T} : \mathfrak{I} \to \mathfrak{O}$, где

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

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

Алгоритмы и разрешимость

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

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

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

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

Алгоритмы[1] придумываются для того,
чтобы **решать** массовые задачи

Алгоритм $\mathcal{A}$ решает задачу $\mathfrak{T} : \mathfrak{I} \to \mathfrak{O}$, 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

M-сводимость

Задача распознавания — это массовая задача
с множеством ответов $\{$**да**, **нет**$\}$ (оно же $\{1, 0\}$, оно же $\{\mathbb{t}, \mathbb{f}\}$)
Задача $\mathfrak{T}_1 : \mathfrak{I}_1 \to \{1, 0\}$ m-сводится (или m-сводима)[1]
к задаче $\mathfrak{T}_2 : \mathfrak{I}_2 \to \{1, 0\}$, если существует алгоритм $\mathcal{A}$, такой что

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

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

Например, если
$\mathfrak{T}_1$ — задача проверки чётности целого числá

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

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

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

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

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

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

M-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 приме́ним $\mathcal{B}$ и $\mathcal{A}$ к входу $\mathfrak{i}$ задачи $\mathfrak{T}_1$ $\quad (\mathfrak{i} \xrightarrow{\mathcal{B}} \mathfrak{j} \xrightarrow{\mathcal{A}} \mathfrak{o})$

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

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

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».

M-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 числá:

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

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

M-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 $\mathfrak{i}$ к своей задаче построить вход $\overline{\mathcal{B}}(\mathfrak{i})$ к моей задаче, такой что

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

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

Математическая логика и логическое программирование

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

Блок 42

Теорема Чёрча

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

ВМК МГУ, 2023/2024, осенний семестр

От МТ к ХЛП

Проблема останова МТ Halt — это следующая проблема распознавания:

- Входы: МТ M, ленточное слово w
- $\mathrm{Halt}(M, w) = \mathbb{t} \Leftrightarrow$ вычисление M на w конечно

Известно, что эта проблема алгоритмически неразрешима

Используя этот факт, покажем неразрешимость проблемы распознавания LogProg, устроенной так:

- Входы: конечная сигнатура σ логики предикатов, ХЛП $\mathcal{P}$ и основной запрос $\mathcal{Q}$ этой сигнатуры
- $\mathrm{LogProg}(\sigma, \mathcal{P}, \mathcal{Q}) = \mathbb{t} \Leftrightarrow \mathcal{Q} \to^*_{\mathcal{P}} \square$

От МТ к ХЛП

Утверждение. **Проблема** Halt m-**сводима к проблеме** LogProg

Доказательство.

В блоке 40 предложен алгоритм $\mathfrak{A}$, для произвольных МТ $M = (\mathcal{A}, \mathbf{\Lambda}, \mathcal{Q}, \mathbf{q}_0, \mathbf{q}_f, \pi)$ и ленточного слова w преобразующий пару (M, w) в конечную сигнатуру, ХЛП и основной запрос $\overline{\mathfrak{A}}(M, w) = (\sigma, \mathcal{P}, \mathcal{Q})$:

- $\sigma = \langle \mathcal{A} \cup \mathcal{Q} \cup \{\mathbf{nil}\}, \{\mathbf{.}^{(2)}\}, \{\mathrm{p}^{(1)}\}\rangle$,
- $\mathcal{P} = \mathcal{P}^1_M$,
- $\mathcal{Q} = ?\mathrm{p}(\tau_{(\mathbf{\Lambda}, \mathbf{q}_0, w\mathbf{\Lambda})})$ —

такие что вычисление M на w конечно $\Leftrightarrow$ $\mathcal{Q} \to^*_{\mathcal{P}} \square$

Значит, $\mathrm{Halt}(M, w) = \mathrm{LogProg}(\overline{\mathfrak{A}}(M, w))$, что и означает m-сводимость Halt к LogProg ▼

Следствие. **Проблема** LogProg **алгоритмически неразрешима**

Доказательство. Достаточно совместить последнее утверждение с известной неразрешимостью проблемы останова МТ и следствием из теоремы об m-сводимости ▼

От ХЛП к логике предикатов (ЛП)

Проблема общезначимости формул ЛП Valid — это проблема распознавания, устроенная так:

- Вход: конечная[1] сигнатура σ и формула ЛП φ этой сигнатуры
- $\mathrm{Valid}(\sigma, \varphi) = \mathrm{tt} \Leftrightarrow \models \varphi$

Утверждение. **Проблема** LogProg m**-сводима к проблеме** Valid

Доказательство.

Достаточно показать, как по ХЛП $\mathcal{P} = \{\mathcal{R}_1, \ldots, \mathcal{R}_k\}$ и основному запросу $\mathcal{Q}$ этой сигнатуры построить формулу φ (*той же конечной сигнатуры, что и программа с запросом*), такую что

$$\mathcal{Q} \to^*_{\mathcal{P}} \square \quad \Leftrightarrow \quad \models \varphi$$

1 Приложив некоторые усилия, можно изложить всё и для сигнатур с бесконечным множеством символов, но для экономии времени не будем этого делать

Утверждение: LogProg $\overset{m}{\mapsto}$ Valid **(доказательство)**

$\mathcal{P} = \{\mathcal{R}_1, \ldots, \mathcal{R}_k\}$ — ХЛП + $\mathcal{Q}$ — основной запрос $\to ?$ φ

$\mathcal{Q} \to^*_{\mathcal{P}} \square$

$\Leftrightarrow$ (по корректности операционной семантики ХЛП)

существует правильный ответ θ на запрос $\mathcal{Q}$ к $\mathcal{P}$

$\Leftrightarrow$ (т.к. $\mathcal{Q}$ — основной запрос)

ε — правильный ответ на запрос $\mathcal{Q}$ к $\mathcal{P}$

$\Leftrightarrow$ (по определению правильного ответа на запрос к ХЛП)

$\{\Phi_{\mathcal{R}_1}, \ldots, \Phi_{\mathcal{R}_k}\} \models \Phi_{\mathcal{Q}}$

$\Leftrightarrow$ (по теореме о логическом следствии)

$\models \Phi_{\mathcal{R}_1} \,\&\, \ldots \,\&\, \Phi_{\mathcal{R}_k} \to \Phi_{\mathcal{Q}}$

Значит, подходящей будет формула $\varphi = \Phi_{\mathcal{R}_1} \,\&\, \ldots \,\&\, \Phi_{\mathcal{R}_k} \to \Phi_{\mathcal{Q}}$ ▼

Теорема Чёрча

Проблема общезначимости формул логики предикатов (Valid**) алгоритмически неразрешима**

Доказательство. Достаточно совместить последнее утверждение, полученную ранее неразрешимость проблемы LogProg и следствие из теоремы об m-сводимости ▼

Следствие. Проблема общезначимости формул логики предикатов сигнатуры $\langle \emptyset, \{\mathbf{f}^{(2)}\}, \{\mathrm{P}^{(1)}\}\rangle$ алгоритмически неразрешима

Доказательство. Достаточно заметить, что

- Для сведéния Halt к LogProg и затем к Valid потребовалась сигнатура с конечным множеством констант $\mathcal{A} \cup \mathcal{Q} \cup \{\mathbf{nil}\}$, одним функциональным символом $\mathbf{.}^{(2)}$ и одним предикатным символом $\mathrm{p}^{(1)}$
- Если в (не)общезначимой формуле заменить константу на переменную, не встречающуюся в формуле, то формула останется (не)общезначимой ▼

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, 2023/2024, 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 числá 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: $\mathrm{plus}(\mathtt{X}, \mathtt{Y}, \mathtt{Z}) = \mathbb{t} \Leftrightarrow \mathtt{Z} = \mathtt{X} + \mathtt{Y}$
3. Реализовать необходимые предикаты для выбранного представления:

$$\mathrm{plus}(\mathbf{nil}, \mathtt{L}, \mathtt{L});$$
$$\mathrm{plus}(\mathtt{L}, \mathbf{nil}, \mathtt{L});$$
$$\mathrm{plus}(\mathbf{0.}\mathtt{L}, \mathtt{X.M}, \mathtt{X.N}) \leftarrow \ldots;$$
$$\mathrm{plus}(\mathbf{1.}\mathtt{L}, \mathtt{X.M}, \mathtt{Y.N}) \leftarrow \ldots;$$
$$\ldots$$

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 обсу́дим тип $\mathrm{integer}$ целых чисел:
$\mathbf{0}, \mathbf{1}, (-\mathbf{1}), \mathbf{2}, (-\mathbf{2}), \ldots$

С точки зрения логики предикатов,

- $\mathbf{0}, \mathbf{1}, (-\mathbf{1}), \mathbf{2}, (-\mathbf{2}), \ldots$ — это целочисленные константы
- $\mathrm{integer}$ — это одноместный предикатный символ: $\mathrm{integer}(\mathtt{x}) =$ «$\mathtt{x}$ — целое число»

Типы данных: integer

integer в логических программах — это встроенный одноместный предикат:

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

Например:

$?\mathrm{integer}(\mathbf{3}), \mathrm{p}(\mathtt{X})$	$?\mathrm{integer}(\mathbf{0.nil})$	$?\mathrm{integer}(\mathbf{1}+\mathbf{2})$	$?\mathrm{integer}(\mathtt{X})$
$\downarrow \varepsilon$	тупик	тупик	тупик
$?\mathrm{p}(\mathtt{X})$			

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 обсу́дим отдельно позже

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

Типы данных: integer, $<$, $\leq$, $>$, $\geq$

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:

$?\mathbf{1} < \mathbf{3}, \mathrm{p}(\mathrm{X})$
$\downarrow \varepsilon$
$?\mathrm{p}(\mathrm{X})$

$?\mathbf{3} < \mathbf{1}$
тупик

$?\mathrm{X} < \mathbf{2}$
тупик

$?\mathbf{1} < \mathbf{1} + \mathbf{2}$
тупик

Как видно по самому правому и самому левому примерам, выражение $\mathbf{1} + \mathbf{2}$ расценивается программой не как число $\mathbf{3}$, 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 $t_1 \,\mathrm{is}\, t_2$:
  - 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:

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

$?\mathbf{3}\,\mathrm{is}\,\mathbf{1}+\mathbf{2}$ тупик

$?\mathtt{X}\,\mathrm{is}\,\mathbf{1}+\mathtt{Y}$ тупик

$?\mathbf{1}+\mathbf{2}\,\mathrm{is}\,\mathbf{3}$ тупик

$?\mathbf{1}+\mathbf{2}\,\mathrm{is}\,\mathtt{X}$ тупик

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 (=, ==, \=, =\=)

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:

$$?\mathtt{X} + \mathbf{1} == \mathtt{X} + \mathbf{1}, \mathrm{p}(\mathtt{X}) \quad \downarrow \varepsilon \quad ?\mathrm{p}(\mathtt{X})$$

$$?\mathtt{X} == \mathbf{1} + \mathbf{2} \quad \text{тупик}$$

$$?\mathbf{3} == \mathbf{1} + \mathbf{2} \quad \text{тупик}$$

$$?\mathtt{X} + \mathbf{1} == \mathbf{1} + \mathtt{X} \quad \text{тупик}$$

$$?\mathtt{X} + \mathbf{1} == \mathbf{2} + \mathtt{X} \quad \text{тупик}$$

$$?\mathbf{1} + \mathbf{2} == \mathbf{2} + \mathbf{1} \quad \text{тупик}$$

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

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

- Критерий выполнимости $t_1 =\backslash= t_2$: t_1 и t_2 не совпадают посимвольно
- Унификатор: ε

Например:

$$?\mathtt{X} + \mathbf{1} =\backslash= \mathtt{X} + \mathbf{1}$$
тупик

$$?\mathtt{X} =\backslash= \mathbf{1} + \mathbf{2}, \mathrm{p}(\mathtt{X})$$
$$\downarrow \varepsilon$$
$$?\mathrm{p}(\mathtt{X})$$

$$?\mathbf{3} =\backslash= \mathbf{1} + \mathbf{2}, \mathrm{p}(\mathtt{X})$$
$$\downarrow \varepsilon$$
$$?\mathrm{p}(\mathtt{X})$$

$$?\mathtt{X} + \mathbf{1} =\backslash= \mathbf{1} + \mathtt{X}, \mathrm{p}(\mathtt{X})$$
$$\downarrow \varepsilon$$
$$?\mathrm{p}(\mathtt{X})$$

$$?\mathtt{X} + \mathbf{1} =\backslash= \mathbf{2} + \mathtt{X}, \mathrm{p}(\mathtt{X})$$
$$\downarrow \varepsilon$$
$$?\mathrm{p}(\mathtt{X})$$

$$?\mathbf{1} + \mathbf{2} =\backslash= \mathbf{2} + \mathbf{1}, \mathrm{p}(\mathtt{X})$$
$$\downarrow \varepsilon$$
$$?\mathrm{p}(\mathtt{X})$$

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

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

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

Например:

$$?\mathtt{X} + \mathbf{1} = \mathtt{X} + \mathbf{1}, \mathrm{p}(\mathtt{X}) \quad \downarrow \varepsilon \quad ?\mathrm{p}(\mathtt{X})$$

$$?\mathtt{X} = \mathbf{1} + \mathbf{2}, \mathrm{p}(\mathtt{X}), \mathrm{r}(\mathtt{Y}) \quad \downarrow \{\mathtt{X}/\mathbf{1} + \mathbf{2}\} \quad ?\mathrm{p}(\mathbf{1} + \mathbf{2}), \mathrm{r}(\mathtt{Y})$$

$$?\mathbf{3} = \mathbf{1} + \mathbf{2}$$
тупик

$$?\mathtt{X} + \mathbf{1} = \mathbf{1} + \mathtt{X}, \mathrm{p}(\mathtt{X}), \mathrm{r}(\mathtt{Y}) \quad \downarrow \{\mathtt{X}/\mathbf{1}\} \quad ?\mathrm{p}(\mathbf{1}), \mathrm{r}(\mathtt{Y})$$

$$?\mathtt{X} + \mathbf{1} = \mathbf{2} + \mathtt{X}$$
тупик

$$?\mathbf{1} + \mathbf{2} = \mathbf{2} + \mathbf{1}$$
тупик

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

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

- Критерий выполнимости $t_1 \backslash = t_2$: $\text{У}(t_1, t_2) = \emptyset$
- Унификатор: ε

Например:

$?\text{X} + \mathbf{1} \backslash = \text{X} + \mathbf{1}$

тупик

$?\text{X} \backslash = \mathbf{1} + \mathbf{2}$

тупик

$?\mathbf{3} \backslash = \mathbf{1} + \mathbf{2}, \text{p}(\text{X})$

$\downarrow \varepsilon$

$?\text{p}(\text{X})$

$?\text{X} + \mathbf{1} \backslash = \mathbf{1} + \text{X}$

тупик

$?\text{X} + \mathbf{1} \backslash = \mathbf{2} + \text{X}, \text{p}(\text{X})$

$\downarrow \varepsilon$

$?\text{p}(\text{X})$

$?\mathbf{1} + \mathbf{2} \backslash = \mathbf{2} + \mathbf{1}, \text{p}(\text{X})$

$\downarrow \varepsilon$

$?\text{p}(\text{X})$

Математическая логика и логическое программирование

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

Блок 44

Логические программы:
стековые вычисления

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

ВМК МГУ, 2023/2024, осенний семестр

С математической точки зрения, в интерпретаторах логических программ обычно используется стандартная стратегия построения дерева вычислений, согласно которой для применения правила

- всегда выбирается самая левая подцель и
- дерево вычислений обходится в глубину согласно порядку правил

Но это только математический взгляд: в интерпретаторах обычно нет структур данных для представления деревьев, и концепция обхода дерева вычислений в глубину реализована в другом эффективном виде

Простейший эффективный способ организации перебора вычислений согласно стандартной стратегии основан на использовании стека (магазина)

Элемент стека соответствует вершине дерева вычислений, обход которой начат, но ещё не завершён

В элементе стека для вершины дерева, отвечающей вычислению $\mathcal{Q}_1 \xrightarrow{\mathcal{R}_1,\theta_1} \ldots \xrightarrow{\mathcal{R}_{k-1},\theta_{k-1}} \mathcal{Q}_k$, сохранена основная информация об этом вычислении, необходимая для обхода и для выдачи ответа:

1. Запрос $\mathcal{Q}_k$
2. Множество целевых переменных $V = \mathrm{Var}_{\mathcal{Q}_1}$
3. Частично вычисленный ответ $(\theta_1 \ldots \theta_{k-1})|_V$
4. Порядковый номер i правила, которое будет применяться к запросу следующим согласно обходу

Элементы стека в иллюстрациях будут перечисляться вертикально сверху вниз, аналогично расположению ярусов дерева вычислений

Пример

$1: \ \mathrm{p}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$

$2: \ \mathrm{p}(\mathtt{X}, \mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$

$3: \mathrm{q}(\mathbf{b}); \qquad 4: \mathrm{r}(\mathbf{c}); \qquad 5: \mathrm{s}(\mathbf{b});$

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$

Начинаем вычисление:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1

Применяем правило 1:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1

Правило 1 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	2

Правило 2 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	3

Пример

$$1: \ \mathrm{p}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$2: \ \mathrm{p}(\mathtt{X}, \mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$3: \mathrm{q}(\mathbf{b}); \qquad 4: \mathrm{r}(\mathbf{c}); \qquad 5: \mathrm{s}(\mathbf{b});$$
$$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$$

Применяем правило 3:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	1

Правило 1 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	2

Правило 2 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	3

Пример

$$1: \ \mathrm{p}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$2: \ \mathrm{p}(\mathtt{X}, \mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$3: \mathrm{q}(\mathbf{b}); \qquad 4: \mathrm{r}(\mathbf{c}); \qquad 5: \mathrm{s}(\mathbf{b});$$
$$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$$

Правило 3 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	4

Применяем правило 4:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	4
$?\mathrm{s}(\mathbf{b})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{c}\}$	1

Правило 1 применить нельзя: ...

Правило 2 применить нельзя: ...

Правило 3 применить нельзя: ...

Пример

$$1: \ \mathrm{p}(\mathtt{X},\mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$2: \ \mathrm{p}(\mathtt{X},\mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$3: \mathrm{q}(\mathbf{b}); \qquad 4: \mathrm{r}(\mathbf{c}); \qquad 5: \mathrm{s}(\mathbf{b});$$
$$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$$

Правило 4 применить нельзя:

$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	4
$?\mathrm{s}(\mathbf{b})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{c}\}$	5

Применяем правило 5:

$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	4
$?\mathrm{s}(\mathbf{b})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{c}\}$	5
$\square$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{c}\}$	1

Выдаём ответ $\{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{c}\}$

Пример

$$1: \ \mathrm{p}(\mathtt{X},\mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$2: \ \mathrm{p}(\mathtt{X},\mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$3: \mathrm{q}(\mathbf{b}); \qquad 4: \mathrm{r}(\mathbf{c}); \qquad 5: \mathrm{s}(\mathbf{b});$$
$$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$$

K □ ничего применить нельзя, откат (извлекаем голову стека, увеличиваем номер команды):

$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	4
$?\mathrm{s}(\mathbf{b})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{c}\}$	6

Правила кончились, откат:

$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	5

Пример

$1: \ \mathrm{p}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$

$2: \ \mathrm{p}(\mathtt{X}, \mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$

$3: \mathrm{q}(\mathbf{b}); \qquad 4: \mathrm{r}(\mathbf{c}); \qquad 5: \mathrm{s}(\mathbf{b});$

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$

Правило 5 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	3
$?\mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathbf{b})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{b}\}$	6

Правила кончились, откат:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	4

Правило 4 применить нельзя: ...

Правило 5 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	1
$?\mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	6

Правила кончились, откат:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	2

Пример

$1: \ \mathrm{p}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$

$2: \ \mathrm{p}(\mathtt{X}, \mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$

$3: \mathrm{q}(\mathbf{b}); \qquad 4: \mathrm{r}(\mathbf{c}); \qquad 5: \mathrm{s}(\mathbf{b});$

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$

Применяем правило 2:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	2
$?\mathrm{r}(\mathtt{X}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{Y}/\mathtt{X}\}$	1

Правило 1 применить нельзя: ...

Правило 2 применить нельзя: ...

Правило 3 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	2
$?\mathrm{r}(\mathtt{X}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{Y}/\mathtt{X}\}$	4

Применяем правило 4:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	2
$?\mathrm{r}(\mathtt{X}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{Y}/\mathtt{X}\}$	4
$?\mathrm{s}(\mathbf{c})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{c}, \mathtt{Y}/\mathbf{c}\}$	1

Пример

$$1: \ \mathrm{p}(\mathtt{X},\mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$2: \ \mathrm{p}(\mathtt{X},\mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$3: \mathrm{q}(\mathbf{b}); \qquad 4: \mathrm{r}(\mathbf{c}); \qquad 5: \mathrm{s}(\mathbf{b});$$
$$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$$

Правило 1 применить нельзя: ...

Правило 2 применить нельзя: ...

Правило 3 применить нельзя: ...

Правило 4 применить нельзя: ...

Правило 5 применить нельзя:

$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	2
$?\mathrm{r}(\mathtt{X}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{Y}/\mathtt{X}\}$	4
$?\mathrm{s}(\mathbf{c})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{X}/\mathbf{c}, \mathtt{Y}/\mathbf{c}\}$	6

Правила кончились, откат:

$?\mathrm{p}(\mathtt{X},\mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	ε	2
$?\mathrm{r}(\mathtt{X}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X},\mathtt{Y}\}$	$\{\mathtt{Y}/\mathtt{X}\}$	5

Пример

$$1: \ \mathrm{p}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathrm{r}(\mathtt{Y});$$
$$2: \ \mathrm{p}(\mathtt{X}, \mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$
$$3: \mathrm{q}(\mathbf{b}); \qquad 4: \mathrm{r}(\mathbf{c}); \qquad 5: \mathrm{s}(\mathbf{b});$$
$$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$$

Правило 5 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	2
$?\mathrm{r}(\mathtt{X}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	$\{\mathtt{Y}/\mathtt{X}\}$	6

Правила кончились, откат:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	3

Правило 3 применить нельзя: ...

Правило 4 применить нельзя: ...

Правило 5 применить нельзя:

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{s}(\mathtt{X})$	$\{\mathtt{X}, \mathtt{Y}\}$	ε	6

Правила кончились, откат

Стек пуст, обход завершён

Такая стековая организация вычислений логических программ используется и на практике, но обычно в более «продвинутом» виде

Например, в начале 1980-х годов исследователь Дэвид Уоррен предложил вычислительную модель (Warren abstract machine), задающую модель памяти, набор стеков для эффективной работы с ней и особую *архитектуру системы команд* для удобной низкоуровневой интерпретации логических программ в предложенной модели и системе стеков

Но знание устройства машины Уоррена в этом курсе не нужно: далее будут обсуждаться предикаты, предназначенные для управления вычислениями программ, и чтобы их понять, достаточно знать, как устроены деревья вычислений и стековые вычисления

Математическая логика и логическое программирование

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

Блок 45

Логические программы:
управление вычислениями,
оператор отсечения

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

ВМК МГУ, 2023/2024, осенний семестр

Управление вычислениями

Два основных способа управления вычислениями логических программ, доступные программисту *согласно тому, что уже рассказано про логические программы*:

1. Выбор порядка программных утверждений
   - Сначала записывать базу индукции (рекурсии), а затем выполнять индуктивный переход (рекурсивный вызов)
   - Сначала решать задачи простыми способами, а при неуспехе переходить к трудным

2. Выбор порядка атомов в телах правил
   - Сначала решать простые подзадачи, а после них сложные
   - Сначала вычислять, и после этого использовать вычисленное

Но этого бывает недостаточно

Управление вычислениями

Например, вычисление программы

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

согласно стандартной стратегии на запросе $?\mathrm{elem}(\mathbf{0}, \mathbf{0.1.0.0.nil})$
(«правда ли, что $\mathbf{0}$ содержится в последовательности $[\mathbf{0}, \mathbf{1}, \mathbf{0}, \mathbf{0}]$»)
устроено так (по строкам сверху вниз, в строке слева направо):

$$?\mathrm{elem}(\mathbf{0}, \mathbf{0.1.0.0.nil}) \xrightarrow{1,\ \{\mathtt{X}'/\mathbf{0}, \mathtt{L}'/\mathbf{1.0.0.nil}\}} \square \quad \text{Ответ: да } (\varepsilon)$$

$$2,\ \{\mathtt{X}'/\mathbf{0}, \mathtt{Y}'/\mathbf{0}, \mathtt{L}'/\mathbf{1.0.0.nil}\}\downarrow$$

$$?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.0.nil})$$

$$2,\ \{\mathtt{X}'/\mathbf{0}, \mathtt{Y}'/\mathbf{1}, \mathtt{L}'/\mathbf{0.0.nil}\}\downarrow$$

$$?\mathrm{elem}(\mathbf{0}, \mathbf{0.0.nil}) \xrightarrow{1,\ \{\mathtt{X}'/\mathbf{0}, \mathtt{L}'/\mathbf{0.nil}\}} \square \quad \text{Ответ: да } (\varepsilon)$$

$$2,\ \{\mathtt{X}'/\mathbf{0}, \mathtt{Y}'/\mathbf{0}, \mathtt{L}'/\mathbf{0.nil}\}\downarrow$$

$$?\mathrm{elem}(\mathbf{0}, \mathbf{0.nil}) \xrightarrow{1,\ \{\mathtt{X}'/\mathbf{0}, \mathtt{L}'/\mathbf{0.nil}\}} \square \quad \text{Ответ: да } (\varepsilon)$$

$$2,\ \{\mathtt{X}'/\mathbf{0}, \mathtt{Y}'/\mathbf{0}, \mathtt{L}'/\mathbf{nil}\}\downarrow$$

$$?\mathrm{elem}(\mathbf{0}, \mathbf{nil})$$

тупик

А нельзя ли сделать так, чтобы обход завершился после первого «да»?

Оператор отсечения (!)

Оператор отсечения (!) — это встроенный 0-местный предикат (*записывающийся без аргументов и без скобок*), предназначенный для особого «отсекания» ветвей дерева вычислений логической программы при использовании стандартной стратегии вычисления

В декларативной семантике ! — это тавтология, формула $\mathbb{t}$

Как встроенный предикат ! всегда выполняется, и всегда с унификатором ε

В операционную семантику оператором ! вносятся и другие изменения

Оператор отсечения (!) и стековые вычисления

В терминах стековых вычислений оператор **!** выполняется так:

- Элемент стека может быть помечен произвольным числом меток отсечения вида $\mathbf{!}_i$, где $i \in \mathbb{N}$
- При добавлении элемента v' в стек с головой v согласно шагу вычисления $\mathcal{Q} \to \mathcal{Q}'$:
  - *По умолчанию* при добавлении элемента в стек все метки $\mathbf{!}_i$ копируются из текущей головы стека
  - Если $\mathcal{Q} = ?\mathbf{!}, B_1, \ldots, B_k$, то из v' удаляется метка $\mathbf{!}_i$ с наибольшим индексом i
  - Если $\mathcal{Q} \xrightarrow{\mathcal{R}} \mathcal{Q}'$ для правила $\mathcal{R}$, в теле которого содержится **!**, то в вершины v и v' добавляется метка $\mathbf{!}_i$ с индексом i, бóльшим всех имеющихся
- При откате с удалением вершины v, содержащей запрос вида $?\mathbf{!}, B_1, \ldots, B_k$, определяется метка $\mathbf{!}_i$ с наибольшим индексом i, и откат продолжается до вершины стека без этой метки

Оператор отсечения (!) и стековые вычисления

Пример

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

$2 : \mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$ $\qquad ?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$

Начинаем вычисление:

$?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$	$\emptyset$	ε	1

Правило 1 применить нельзя:

$?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$	$\emptyset$	ε	2

Применяем правило 2:

$?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$	$\emptyset$	ε	2
$?\mathrm{elem}(\mathbf{0}, \mathbf{0.1.0.nil})$	$\emptyset$	ε	1

Применяем правило 1:

$?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$	$\emptyset$	ε	2	
$?\mathrm{elem}(\mathbf{0}, \mathbf{0.1.0.nil})$	$\emptyset$	ε	1	$\mathbf{!}_1$
$?\mathbf{!}$	$\emptyset$	ε	1	$\mathbf{!}_1$

Оператор отсечения (!) и стековые вычисления

Пример

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

$2 : \mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$ $\quad ?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$

Выполняем **!**:

$?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$	$\emptyset$	ε	2	
$?\mathrm{elem}(\mathbf{0}, \mathbf{0.1.0.nil})$	$\emptyset$	ε	1	$\mathbf{!}_1$
?!	$\emptyset$	ε	1	$\mathbf{!}_1$
$\square$	$\emptyset$	ε	1	

Выдаём ответ ε («да»)

Откат:

$?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$	$\emptyset$	ε	2	
$?\mathrm{elem}(\mathbf{0}, \mathbf{0.1.0.nil})$	$\emptyset$	ε	1	$\mathbf{!}_1$
?!	$\emptyset$	ε	2	$\mathbf{!}_1$

Левая подцель — **!**, продолжаем откат, пока не устраним метку $\mathbf{!}_1$:

$?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$	$\emptyset$	ε	3	

Правила кончились, откат

Стек пуст, обход завершён

Оператор отсечения (!) и деревья вычислений

Эффект отсечения можно достаточно наглядно представить и в терминах дерева вычислений:

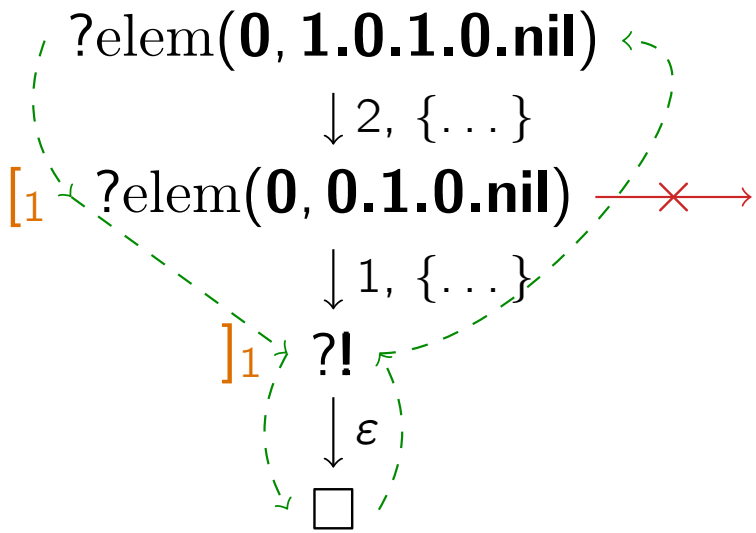
- При обходе дугй $\mathcal{Q}_1 \xrightarrow{\mathcal{R}} \mathcal{Q}_2$, такой что в $\mathcal{R}$ содержится **!**, вершина $\mathcal{Q}_1$ (начало дугй) помечается меткой $[_i$ с уникальным индексом i
  - Это место первой записи метки отсечения в стековом вычислении
- Когда знак **!**, появившийся при обходе дуги, упомянутой выше, становится левой подцелью, этот запрос помечается меткой $]_i$ для того же индекса i
  - Это место последней записи метки отсечения в стековом вычислении
- При возврате в вершину с меткой $]_i$ выполняется перенаправление в родителя вершины с меткой $[_i$ (если родителя нет, то обход завершён)
  - То есть отсекаются все ветви дерева вычислений, которые вырастали бы в дальнейшем обходе из вершин от $[_i$ до $]_i$ включительно

Оператор отсечения (!) и деревья вычислений

Пример

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

$2 : \mathrm{elem}(\mathtt{X}, \mathtt{Y.L}) \leftarrow \mathrm{elem}(\mathtt{X}, \mathtt{L});$ $\quad ?\mathrm{elem}(\mathbf{0}, \mathbf{1.0.1.0.nil})$



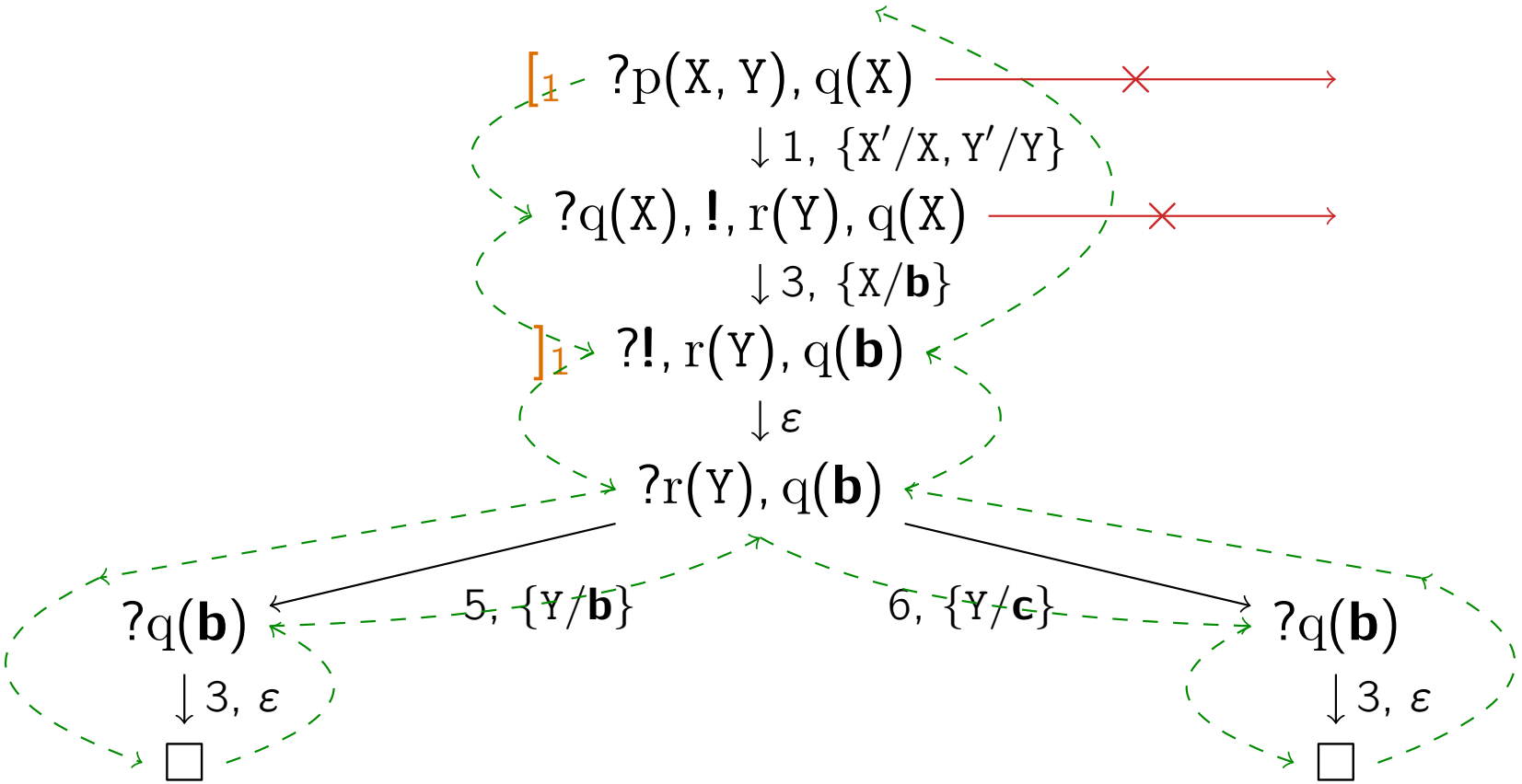
Оператор отсечения (!) и деревья вычислений

Другой пример

$1 : \mathrm{p}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathrm{q}(\mathtt{X}), \mathbf{!}, \mathrm{r}(\mathtt{Y});$ $\quad 2 : \mathrm{p}(\mathtt{X}, \mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$

$3 : \mathrm{q}(\mathbf{b});$ $\quad 4 : \mathrm{q}(\mathbf{c});$ $\quad 5 : \mathrm{r}(\mathbf{b});$ $\quad 6 : \mathrm{r}(\mathbf{c});$

$?\mathrm{p}(\mathtt{X}, \mathtt{Y}), \mathrm{q}(\mathtt{X})$



Ответ: $\{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{b}\}$

Ответ: $\{\mathtt{X}/\mathbf{b}, \mathtt{Y}/\mathbf{c}\}$

Оператор отсечения (!) и деревья вычислений

Правило вида $A \leftarrow B_1, \ldots, B_k, \mathbf{!}, C_1, \ldots, C_m$ можно трактовать так: чтобы решить задачу A, следует проверить, верны ли условия $B_1, \ldots, B_k$ (найти первое попавшееся совокупное решение подзадач $B_1, \ldots, B_k$, если оно есть), и

- если верны (решение найдено), то найти все решения подзадач $C_1, \ldots, C_m$ и не решать A другими способами,
- а если нет, то решить A другими способами

Отталкиваясь от этой трактовки, можно использовать **!** для моделирования конструкций императивной парадигмы:

- Ветвление A : **if** B **then** C **else** D **fi**
  $A \leftarrow B, \mathbf{!}, C;$
  $A \leftarrow D;$
- Цикл A : **while** B **do** C **od**
  $A \leftarrow B, \mathbf{!}, C, A;$
  $A;$

Оператор отсечения (!) и деревья вычислений

Следует иметь в виду, что для ХЛП с оператором отсечения перестаёт быть справедливой теорема о полноте, даже если дерево вычислений конечно: не для всякого правильного ответа можно вычислить какое-либо обобщение

Например, для программы

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

и запроса

$$?\mathrm{elem}(\mathtt{X}, \mathbf{0.1.nil})$$

правильными ответами являются подстановки

$$\{\mathtt{X}/\mathbf{0}\} \quad \text{и} \quad \{\mathtt{X}/\mathbf{1}\},$$

а вычислен будет только ответ

$$\{\mathtt{X}/\mathbf{0}\}$$

Поэтому оператор **!** следует использовать осторожно и только с хорошим пониманием того, какие ветви дерева вычислений будут отсечены и как это в целом повлияет на вычисление ответов

Математическая логика и логическое программирование

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

Блок 46

Отрицание в логическом программировании
Допущение замкнутости мира

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

ВМК МГУ, 2023/2024, осенний семестр

Пример (ХЛП $\mathcal{P}$)

$$\text{птица}(\textbf{орёл}); \qquad \text{летает}(\textbf{орёл});$$
$$\text{птица}(\textbf{пингвин}); \qquad \text{летает}(\textbf{самолёт});$$

Послав подходящий запрос, можно узнать у $\mathcal{P}$, какие по её мнению есть летающие птицы:

$$?\text{птица}(\mathtt{X}), \text{летает}(\mathtt{X})$$

На этот запрос есть ровно один правильный (и он же ровно один SLD-вычислимый) ответ: летающая птица — это орёл

$$\{\mathtt{X}/\textbf{орёл}\}$$

Если бы в телах можно можно было использовать не только атомы, но и их отрицания, то можно было бы естественно записать запрос о нелетающей птице:

$$?\text{птица}(\mathtt{X}), \neg\text{летает}(\mathtt{X})$$

$\neg$ — полезная логическая связка, и хотелось бы её использовать в логических программах

А почему до сих пор было запрещено так использовать $\neg$?

Пример (ХЛП $\mathcal{P}$ и запрос $\mathcal{Q}$)

$$\text{птица}(\textbf{орёл}); \qquad \text{летает}(\textbf{орёл});$$
$$\text{птица}(\textbf{пингвин}); \qquad \text{летает}(\textbf{самолёт});$$
$$?\text{птица}(\texttt{X}), \neg\text{летает}(\texttt{X})$$

Применив любой известный метод проверки общезначимости формул логики предикатов, можно легко убедиться, что

$$\not\models \quad \text{птица}(\textbf{орёл}) \,\&\, \text{птица}(\textbf{пингвин}) \,\&\, \text{летает}(\textbf{орёл}) \,\&\, \text{летает}(\textbf{самолёт}) \to \text{птица}(\textbf{пингвин}) \,\&\, \neg\text{летает}(\textbf{пингвин})$$

А значит (по теореме о логическом следствии),

$$\left\{\begin{matrix}\text{птица}(\textbf{орёл}), \ \text{птица}(\textbf{пингвин}), \\ \text{летает}(\textbf{орёл}), \text{летает}(\textbf{самолёт})\end{matrix}\right\} \not\models \text{птица}(\textbf{пингвин}) \,\&\, \neg\text{летает}(\textbf{пингвин})$$

То есть программа $\mathcal{P}$ не считает пингвина нелетающей птицей

И летающей птицей программа $\mathcal{P}$ пингвина тоже не считает:

$$\left\{\begin{matrix}\text{птица}(\textbf{орёл}), \ \text{птица}(\textbf{пингвин}), \\ \text{летает}(\textbf{орёл}), \text{летает}(\textbf{самолёт})\end{matrix}\right\} \not\models \text{птица}(\textbf{пингвин}) \,\&\, \text{летает}(\textbf{пингвин})$$

При этом $\mathcal{P}$ уверена, что пингвин — это птица:

$$\left\{\begin{matrix}\text{птица}(\textbf{орёл}), \ \text{птица}(\textbf{пингвин}), \\ \text{летает}(\textbf{орёл}), \text{летает}(\textbf{самолёт})\end{matrix}\right\} \models \text{птица}(\textbf{пингвин})$$

Всё-таки летает ли пингвин?

Пример (ХЛП $\mathcal{P}$ и запрос $\mathcal{Q}$)

$$\text{птица}(\textbf{орёл}); \qquad \text{летает}(\textbf{орёл});$$
$$\text{птица}(\textbf{пингвин}); \qquad \text{летает}(\textbf{самолёт});$$
$$?\text{птица}(\mathtt{X}), \neg\text{летает}(\mathtt{X})$$

У $\mathcal{P}$ как у системы формул есть модель, в которой утверждается, что пингвин не летает

Значит, достоверно заключить, что пингвин летает, $\mathcal{P}$ не может

Но у $\mathcal{P}$ есть и модель, в которой утверждается, что пингвин летает

Например, такой моделью $\mathcal{P}$ является $\mathcal{H}$-интерпретация

$$\left\{\begin{array}{l}\text{птица}(\textbf{орёл}),\ \text{птица}(\textbf{пингвин}), \\ \text{летает}(\textbf{орёл}), \text{летает}(\textbf{самолёт}), \text{летает}(\textbf{пингвин})\end{array}\right\}$$

Значит, достоверно заключить, что пингвин не летает, $\mathcal{P}$ тоже не может

Строго говоря, ответ «пингвин — это нелетающая птица» и должен быть неправильным: если невозможно доказать, что пингвин не летает, то из этого не следует, что пингвин летает

Пример (ХЛП $\mathcal{P}$ и запрос $\mathcal{Q}$)

$$\text{птица}(\textbf{орёл}); \qquad \text{летает}(\textbf{орёл});$$
$$\text{птица}(\textbf{пингвин}); \qquad \text{летает}(\textbf{самолёт});$$
$$?\text{птица}(\texttt{X}), \neg\text{летает}(\texttt{X})$$

Можно заставить программиста **явно** записать факт

$$\neg\text{летает}(\textbf{пингвин})$$

Но тогда программисту придётся аналогично перечислить всё то, что, согласно его знаниям, не умеет летать (*и это ОЧЕНЬ много фактов*)

Чтобы программист был освобождён от бремени перечисления всего того, чего нет, принято использовать допущение замкнутости мира (предположение о замкнутости мира; closed world assumption; CWA), имеющее несколько строгих формулировок с таким общим смыслом: **если утверждение невозможно доказать, то оно считается ложным**

Пример (ХЛП $\mathcal{P}$ и запрос $\mathcal{Q}$)

$$\text{птица}(\textbf{орёл}); \qquad \text{летает}(\textbf{орёл});$$
$$\text{птица}(\textbf{пингвин}); \qquad \text{летает}(\textbf{самолёт});$$
$$?\text{птица}(\mathtt{X}), \neg\text{летает}(\mathtt{X})$$

Аналогичный принцип (*презумпция невиновности*) используется в суде: обвиняемый не обязан явно опровергать весь спектр гипотетически возможных фактов, которые могли бы доказать его вину, и считается невиновным, если невозможно доказать вину

Если программа $\mathcal{P}$ (*обвинитель*) не может доказать, что пингвин (*виновен в том, что*) летает, то он не летает (*невиновен*)

В этом смысле ($\models_{cwa}$) пингвин — нелетающая птица:

$$\left\{\begin{matrix}\text{птица}(\textbf{орёл}),\ \text{птица}(\textbf{пингвин}),\\ \text{летает}(\textbf{орёл}),\ \text{летает}(\textbf{самолёт})\end{matrix}\right\} \models_{cwa} \text{птица}(\textbf{пингвин}) \,\&\, \neg\text{летает}(\textbf{пингвин})$$

Попробуем предоставить хотя бы один вариант строгой корректной формулировки CWA

Моделью хорновской логической программы $\mathcal{P}$ будем называть модель системы формул $\Phi_{\mathcal{P}}$

Для семейства $\mathfrak{I}$ $\mathcal{H}$-интерпретаций записью $\cap\mathfrak{I}$ будем обозначать $\mathcal{H}$-интерпретацию $\{A \mid \forall \mathcal{I} \in \mathfrak{I} : A \in I\}$

Лемма (о пересечении моделей ХЛП). Для любой ХЛП $\mathcal{P}$ и любого семейства $\mathfrak{I}$ её эрбрановских моделей интерпретация $\cap\mathfrak{I}$ также является моделью ХЛП $\mathcal{P}$

Доказательство.

Предположим *от противного*, что $\cap\mathfrak{I}$ не является моделью $\mathcal{P}$

Тогда существует правило $\forall \ldots (B_1 \,\&\, \ldots \,\&\, B_k \to A) \in \Phi_{\mathcal{P}}$, такое что $\cap\mathfrak{I} \not\models \forall \ldots (B_1 \,\&\, \ldots \,\&\, B_k \to A)$

По семантике $\forall$, существует набор предметов $\widetilde{d}^n$, такой что $\cap\mathfrak{I} \not\models (B_1 \,\&\, \ldots \,\&\, B_k \to A)[\widetilde{d}^n]$

Для технической простоты *без ограничения общности* положим, что в сигнатуре языка формул содержится хотя бы одна константа

Лемма о пересечении моделей ХЛП (доказательство)

$$(\forall \ldots (B_1 \,\&\, \ldots \,\&\, B_k \to A)) \in \Phi_{\mathcal{P}}$$
$$\cap\mathfrak{I} \not\models (B_1 \,\&\, \ldots \,\&\, B_k \to A)[\widetilde{d}^n] \text{ для любых } \widetilde{d}^n$$

Тогда предметами $\mathcal{H}$-интерпретаций являются основные термы, и последнее соотношение означает, что существует основной пример $B_1^g \,\&\, \ldots \,\&\, B_k^g \to A^g$ правила из $\Phi_{\mathcal{P}}$, такой что $\cap\mathfrak{I} \not\models B_1^g \,\&\, \ldots \,\&\, B_k^g \to A^g$

По семантике $\&$ и $\to$: $\cap\mathfrak{I} \models B_1^g$, ..., $\cap\mathfrak{I} \models B_k^g$, $\cap\mathfrak{I} \not\models A^g$

По теоретико-множественному заданию $\mathcal{H}$-интерпретаций:

$$B_1^g \in \cap\mathfrak{I}, \quad \ldots, \quad B_k^g \in \cap\mathfrak{I}, \quad A^g \notin \cap\mathfrak{I}$$

По определению $\cap\mathfrak{I}$:

- существует $\mathcal{H}$-модель $\mathcal{I}$ ХЛП $\mathcal{P}$, такая что $A^g \notin \mathcal{I}$, и
- для любой $\mathcal{H}$-модели $\mathcal{J}$ ХЛП $\mathcal{P}$ (в том числе для $\mathcal{I}$) верно $B_1^g \in \mathcal{J}$, ..., $B_k^g \in \mathcal{J}$

Применив теоретико-множественный способ задания $\mathcal{H}$-интерпретаций и семантику $\&$, $\to$ и $\forall$ к $\mathcal{I}$ так же, как они применялись к $\cap\mathfrak{I}$, но в обратном порядке, получим соотношение $\mathcal{I} \not\models \forall \ldots (B_1 \,\&\, \ldots \,\&\, B_k \to A)$

То есть $\mathcal{I}$ не является моделью для $\Phi_{\mathcal{P}}$, что *противоречит* выбору $\mathcal{I}$ ▼

Наименьшей моделью ХЛП $\mathcal{P}$ будем называть $\mathcal{H}$-модель, наименьшую относительно теоретико-множественного включения среди всех $\mathcal{H}$-моделей ХЛП $\mathcal{P}$ (*то есть модель, вложенную как множество в каждую другую модель*)

Теорема. Для любой ХЛП $\mathcal{P}$ существует наименьшая модель

Доказательство.

По последней лемме, если $\mathfrak{I}$ — семейство всех $\mathcal{H}$-моделей ХЛП $\mathcal{P}$, то $\cap\mathfrak{I}$ — $\mathcal{H}$-модель ХЛП $\mathcal{P}$

Нетрудно видеть, что эта модель является наименьшей по теоретико-множественному включению в семействе $\mathfrak{I}$ ▼

$M_{\mathcal{P}}$ — так будем обозначать наименьшую модель ХЛП $\mathcal{P}$

Теорема. $M_{\mathcal{P}} = B_{\mathcal{H}} \cap \{\varphi \mid \Phi_{\mathcal{P}} \models \varphi\}$

Доказательство. (*по прежнему считаем, что в сигнатуре есть хотя бы одна константа*)

($\supseteq$): Рассмотрим основной атом A, такой что $\Phi_{\mathcal{P}} \models A$

По определению логического следования, верно $M_{\mathcal{P}} \models A$

Следовательно, $A \in M_{\mathcal{P}}$

($\subseteq$): Предположим *от противного*, что существует основной атом A, такой что $A \in M_{\mathcal{P}}$ и $\Phi_{\mathcal{P}} \not\models A$

По определению логического следования, существует модель $\mathcal{I}$ системы $\Phi_{\mathcal{P}}$, такая что $\mathcal{I} \not\models A$

По семантике $\neg$, верно $\mathcal{I} \models \neg A$

Значит, верно и $\mathcal{I} \models \Phi_{\mathcal{P}} \cup \{\neg A\}$

По теореме об $\mathcal{H}$-интерпретациях, существует эрбрановская интерпретация $\mathcal{I}_h$, такая что $\mathcal{I}_h \models \Phi_{\mathcal{P}} \cup \{\neg A\}$

По семантике $\neg$, верно и $\mathcal{I}_h \not\models A$

По теоретико-множественному заданию $\mathcal{H}$-интерпретаций, верно $A \notin \mathcal{I}_h$

Из $A \notin \mathcal{I}_h$ и $\mathcal{I}_h \models \Phi_{\mathcal{P}}$ следует $A \notin M_{\mathcal{P}}$, что *противоречит* выбору A ▼

Пример

Для программы

$$\begin{array}{ll} \text{птица}(\textbf{орёл}); & \text{летает}(\textbf{орёл}); \\ \text{птица}(\textbf{пингвин}); & \text{летает}(\textbf{самолёт}); \\ \text{обтекаемый}(\textbf{самолёт}); & \\ \text{обтекаемый}(\mathtt{X}) \leftarrow \text{птица}(\mathtt{X}), \text{летает}(\mathtt{X}); & \end{array}$$

наименьшая модель устроена так:

$$\left\{ \begin{array}{ll} \text{птица}(\textbf{орёл}), & \text{птица}(\textbf{пингвин}), \\ \text{летает}(\textbf{орёл}), & \text{летает}(\textbf{самолёт}), \\ \text{обтекаемый}(\textbf{орёл}), & \text{обтекаемый}(\textbf{самолёт}) \end{array} \right\}$$

В этой модели содержится наименьший набор основных фактов, которые могут быть истинными согласно написанному в программе

Будем считать, что формула φ следует в допущении замкнутости мира из системы $\Phi_{\mathcal{P}}$, отвечающей ХЛП $\mathcal{P}$ ($\Phi_{\mathcal{P}} \models_{cwa} \varphi$), если $M_{\mathcal{P}} \models \varphi$

В частности, в таком определении действительно верно, что пингвин — нелетающая птица:

$$\left\{\begin{matrix}\text{птица}(\textbf{орёл}),\ \text{птица}(\textbf{пингвин}),\\ \text{летает}(\textbf{орёл}),\ \text{летает}(\textbf{самолёт})\end{matrix}\right\} \models_{cwa} \text{птица}(\textbf{пингвин}) \,\&\, \neg\text{летает}(\textbf{пингвин}),$$

так как в наименьшей модели программы содержится (выполняется) атом «$\text{птица}(\textbf{пингвин})$» и не содержится (не выполняется) атом «$\text{летает}(\textbf{пингвин})$»

Математическая логика и логическое программирование

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

Блок 47

Логические программы:
оператор отрицания,
SLDNF-резолюция

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

ВМК МГУ, 2023/2024, осенний семестр

Оператор отрицания (not)

Оператор отрицания **not** — это особый встроенный оператор, устроенный так:

- **not** — это не предикатный символ, не функциональный символ и не константа
- На месте атома в запросе или в теле правила может быть записано выражение **not**(A),[1] где A — это атом
- В декларативной семантике **not**(A) отвечает формуле $\neg A$ в допущении замкнутости мира

А с операционной семантикой оператора **not** попробуем разобраться отдельно

Чтобы не перегружать материал тем, что относится к общему случаю и представляет в основном теоретический интерес, ограничимся семантикой **not** для стандартной стратегии вычисления

1 В Prolog это записывается так: «\+ A»

Оператор отрицания (not)

Хотелось бы устроить вычисление логической программы $\mathcal{P}$ так, чтобы для подцели **not**(A) интерпретатор программ *хотя бы для основных атомов* A продолжал вычисление, если $\Phi_{\mathcal{P}} \models_{cwa} \neg A$, а иначе констатировал тупик

Проверка этого соотношения для ХЛП $\mathcal{P}$ равносильна проверке того, что не существует ни одного успешного вычисления $\mathcal{P}$

При обсуждении теоремы Чёрча был показано, что задача такой проверки алгоритмически неразрешима (и тем более неразрешима для ХЛП с оператором **not**)

Значит, в полной мере и эффективно воплотить допущение замкнутости мира в операционной семантике логических программ не получится

Придётся пойти на компромисс и учесть оператор **not** «достаточно разумно», сохранив эффективность вычислений

SLDNF-резолюция

Правило SLDNF-резолюции (SLD with Negation as Failure) — это аналог правила SLD-резолюции, использующийся для логических программ с отрицанием

Есть несколько строгих формулировок этого правила, и для примера обсудим формулировку, наиболее приближенную к «реальному» вычислению логических программ согласно стандартной стратегии

Как и для встроенных предикатов, семантику **not** можно задать как сочетание критерия выполнимости и унификатора:

- Критерий выполнимости **not**(A): строится (обходится) дерево вычислений для запроса ?A (вспомогательное дерево), и обход этого дерева вполне неуспешен согласно изложенному далее
- Унификатор: ε

Остальные понятия для вычислений программ (успешное вычисление, результат вычисления, вычислимый ответ, дерево вычислений) вводятся так же, как для «SLD», с заменой «SLD» на «SLDNF»

SLDNF-резолюция

При построении вспомогательного дерева возможны три исхода:

1. В процессе обхода дерева получен $\square$
   - Обнаружены успешное вычисление и вычисленный им ответ
   - В этом случае обход дерева успешен
2. Дерево построено, оно конечно, и в нём нет ни одного $\square$
   - По итогам обхода не обнаружено ни одно успешное вычисление
   - В этом случае обход дерева вполне неуспешен
3. Дерево и возникающие вспомогательные деревья обходятся бесконечно, и в строящемся фрагменте дерева нет ни одного $\square$
   - Значит, программа «зациклилась» при проверке вполне-неуспешности дерева
   - В этом случае обход дерева неуспешен, но *не вполне*
   - Соответствующее вычисление программы (для исходного запроса) признаётся бесконечным («сингулярная бесконечность»), и обход исходного дерева и всех остальных вспомогательных поддеревьев немедленно завершается

SLDNF-резолюция

Теорема (о корректности SLDNF-резолюции). Для любой ХЛП с операторами отрицания $\mathcal{P}$ и любого запроса с операторами отрицания $\mathcal{Q}$ верно следующее: результат любого успешного SLDNF-резолютивного вычисления $\mathcal{P}$ на $\mathcal{Q}$ является правильным ответом в допущении замкнутости мира

Доказательство опустим: времени потратим много, а пользы будет мало

А аналогичная теорема о полноте оказывается неверной из-за

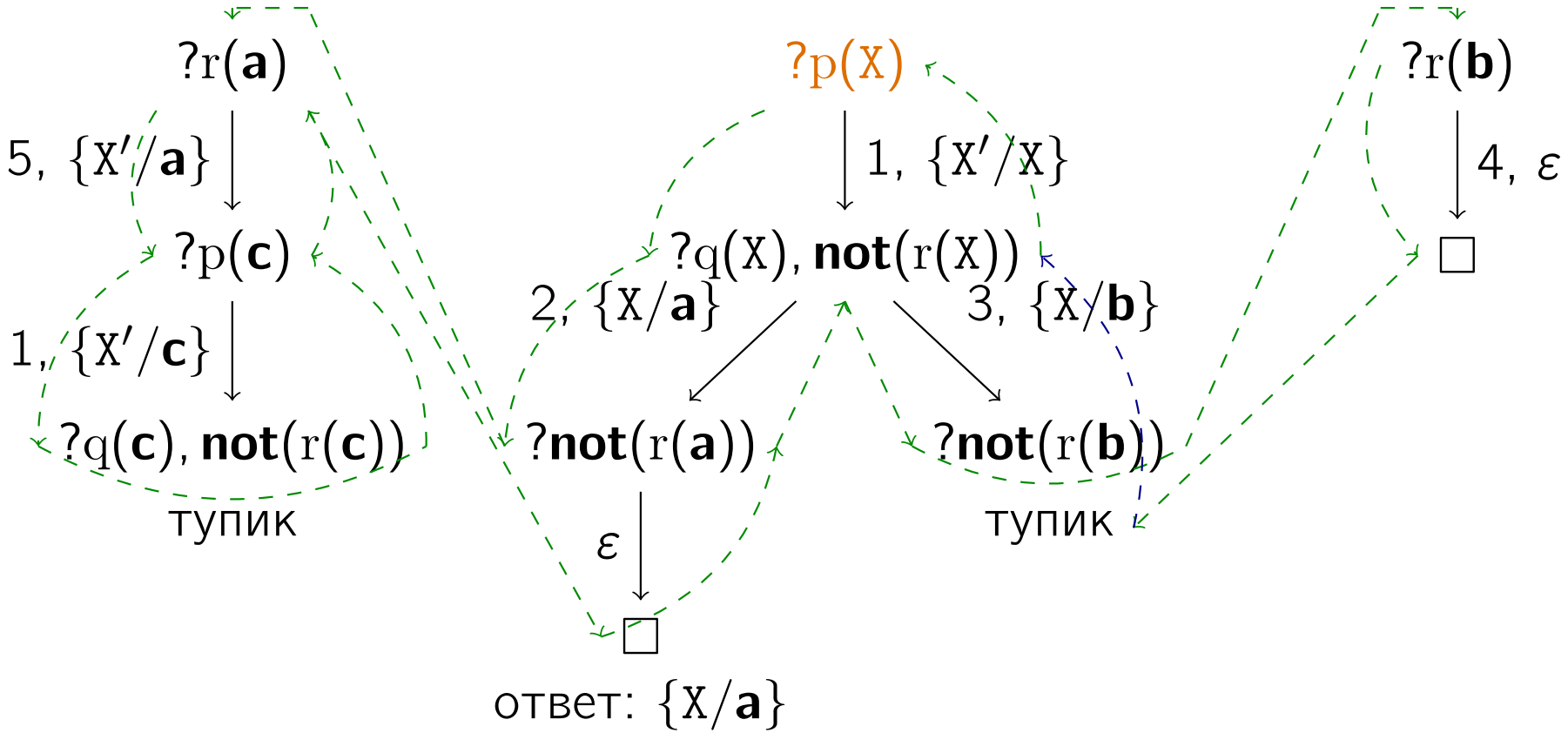
- использования стандартной стратегии вычисления (*этого можно было бы избежать*) и
- возможного появления бесконечных вспомогательных деревьев (*а этого избежать в общем случае не выйдет*)

SLDNF-резолюция

Примеры

$$1: \ \mathrm{p}(\mathrm{X}) \leftarrow \mathrm{q}(\mathrm{X}), \mathbf{not}(\mathrm{r}(\mathrm{X}));$$
$$2: \mathrm{q}(\mathbf{a}); \quad 3: \mathrm{q}(\mathbf{b});$$
$$4: \mathrm{r}(\mathbf{b}); \quad 5: \mathrm{r}(\mathrm{X}) \leftarrow \mathrm{p}(\mathbf{c});$$

Дерево SLDNF-резолютивных вычислений этой программы для запроса $?\mathrm{p}(\mathrm{X})$ и стандартной стратегии вычисления:

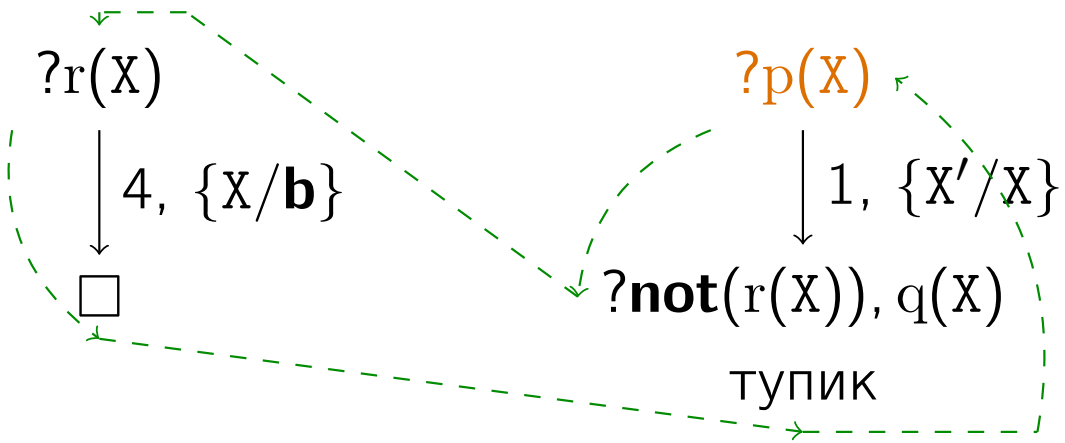


SLDNF-резолюция

Примеры

$$1: \ \mathrm{p}(\mathtt{X}) \leftarrow \mathbf{not}(\mathrm{r}(\mathtt{X})), \mathrm{q}(\mathtt{X});$$
$$2: \mathrm{q}(\mathbf{a}); \quad 3: \mathrm{q}(\mathbf{b});$$
$$4: \mathrm{r}(\mathbf{b}); \quad 5: \mathrm{r}(\mathtt{X}) \leftarrow \mathrm{p}(\mathbf{c});$$

Дерево SLDNF-резолютивных вычислений этой программы для запроса $?\mathrm{p}(\mathtt{X})$ и стандартной стратегии вычисления:

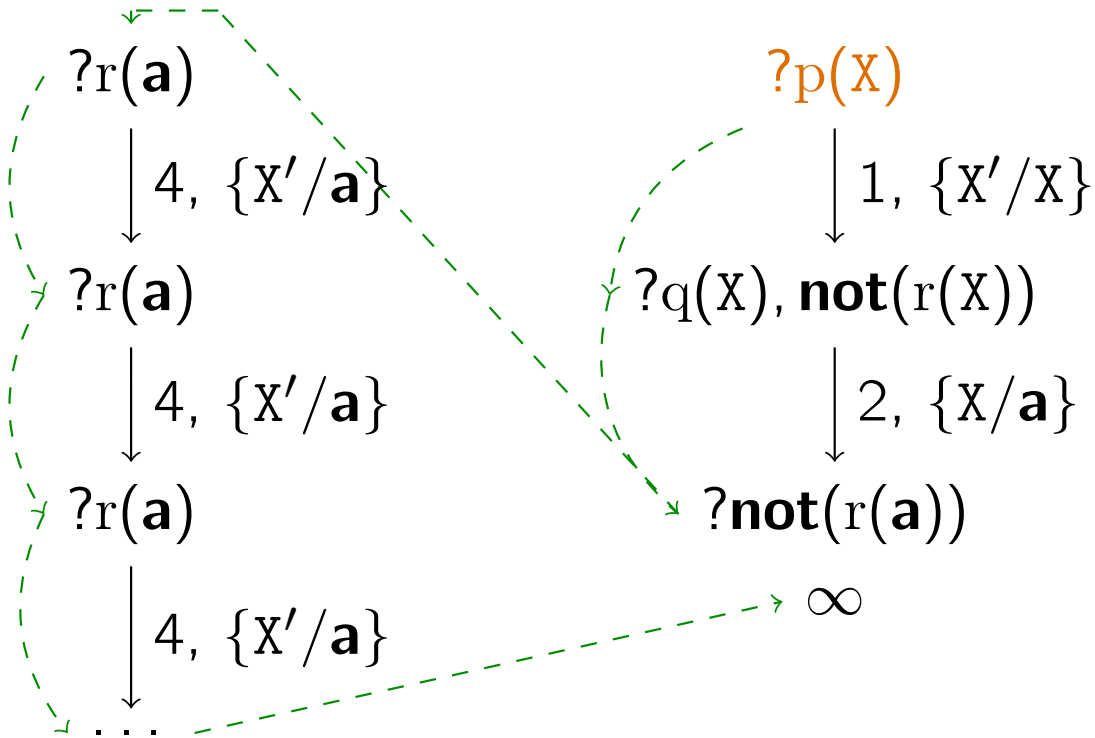


SLDNF-резолюция

Примеры

$$1: \ \mathrm{p}(\mathtt{X}) \leftarrow \mathrm{q}(\mathtt{X}), \mathbf{not}(\mathrm{r}(\mathtt{X}));$$
$$2: \mathrm{q}(\mathbf{a}); \quad 3: \mathrm{q}(\mathbf{b});$$
$$4: \ \mathrm{r}(\mathtt{X}) \leftarrow \mathrm{r}(\mathtt{X});$$

Дерево SLDNF-резолютивных вычислений этой программы для запроса $?\mathrm{p}(\mathtt{X})$ и стандартной стратегии вычисления:

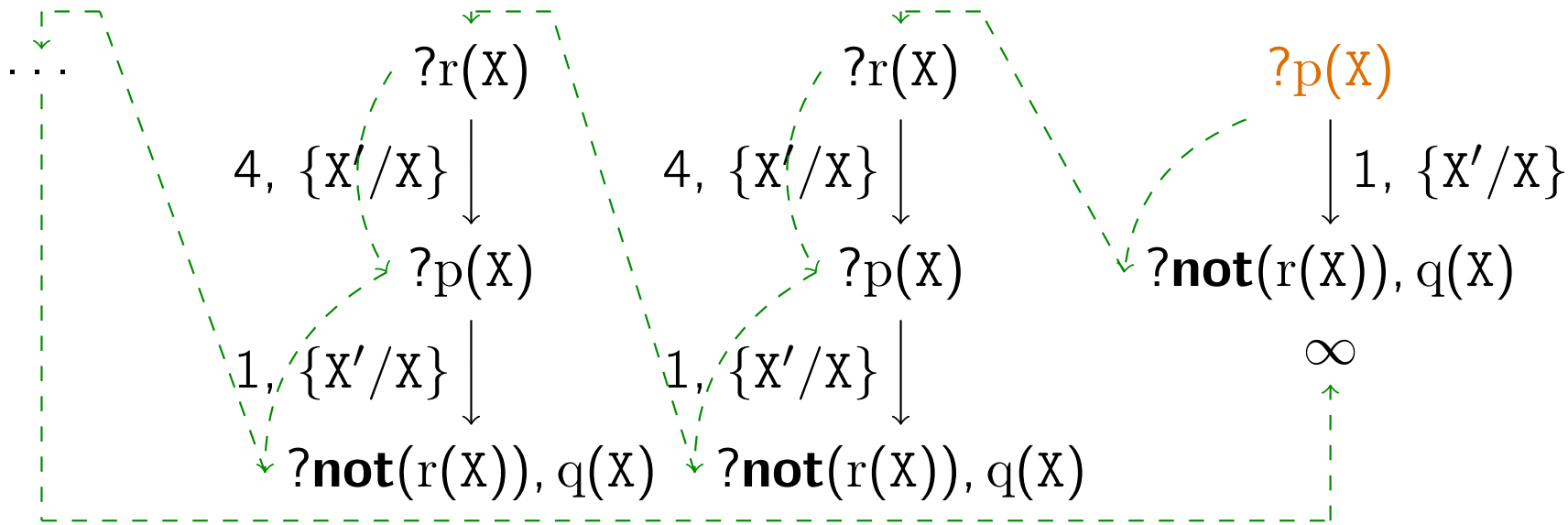


SLDNF-резолюция

Примеры

$$1: \ \mathrm{p}(\mathtt{X}) \leftarrow \mathbf{not}(\mathrm{r}(\mathtt{X})), \mathrm{q}(\mathtt{X});$$
$$2: \mathrm{q}(\mathbf{a}); \quad 3: \mathrm{q}(\mathbf{b});$$
$$4: \ \mathrm{r}(\mathtt{X}) \leftarrow \mathrm{p}(\mathtt{X});$$

Дерево SLDNF-резолютивных вычислений этой программы для запроса $?\mathrm{p}(\mathtt{X})$ и стандартной стратегии вычисления:



SLDNF-резолюция

Более «осмысленный» пример

Логическая программа, такая что $\mathrm{p}(\mathtt{X}, \mathtt{L}, \mathtt{M})$ вычисляет в $\mathtt{X}$ произвольный элемент списка $\mathtt{L}$, не содержащийся в списке $\mathtt{M}$:

$$1 : \mathrm{p}(\mathtt{X}, \mathtt{L}, \mathtt{M}) \leftarrow \mathrm{e}(\mathtt{X}, \mathtt{L}), \textbf{not}(\mathrm{e}(\mathtt{X}, \mathtt{M}));$$
$$2 : \mathrm{e}(\mathtt{X}, \mathtt{X}.\mathtt{L});$$
$$3 : \mathrm{e}(\mathtt{X}, \mathtt{Y}.\mathtt{L}) \leftarrow \mathrm{e}(\mathtt{X}, \mathtt{L});$$

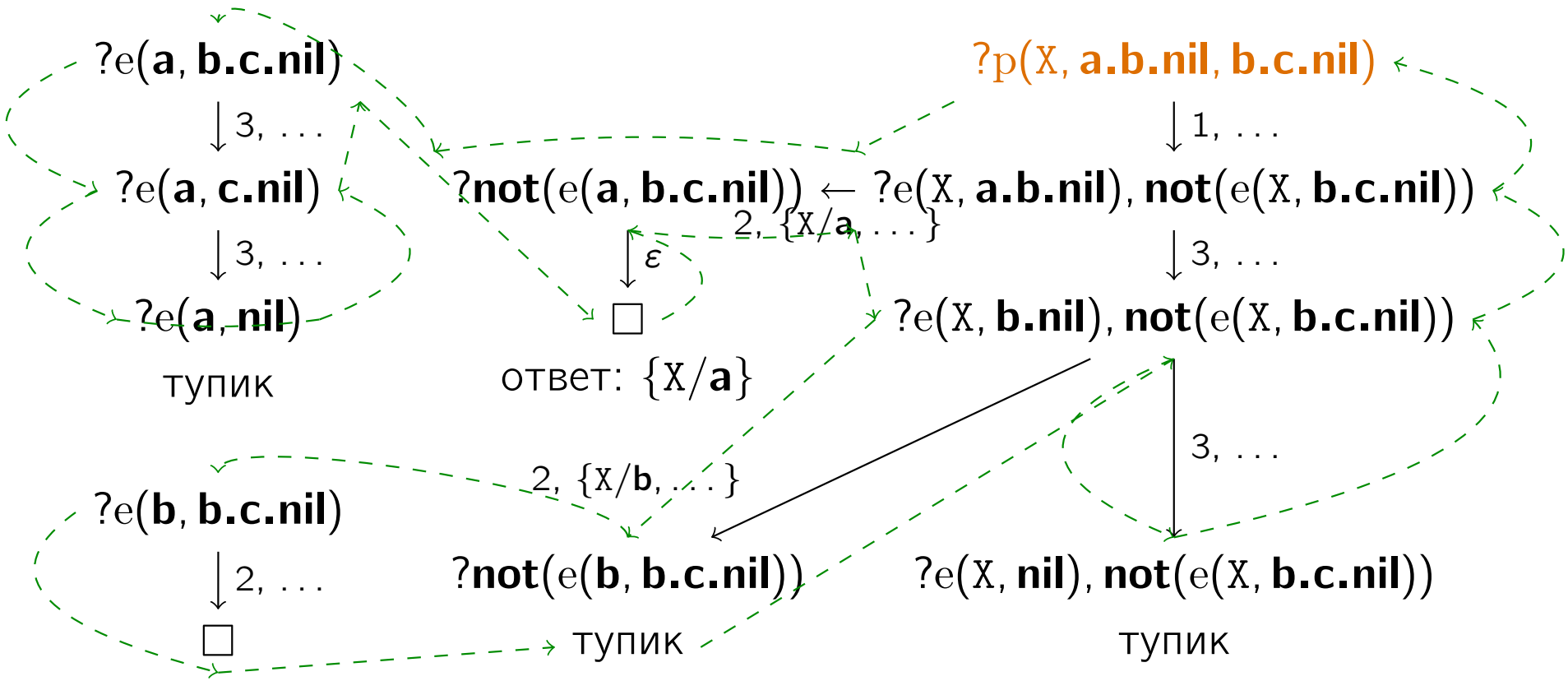
Единственный правильный ответ на запрос $?\mathrm{p}(\mathtt{X}, \textbf{a.b.nil}, \textbf{b.c.nil})$ в допущении замкнутости мира: $\{\mathtt{X}/\textbf{a}\}$

SLDNF-резолюция

Более «осмысленный» пример

$$1: \ \mathrm{p}(\mathtt{X}, \mathtt{L}, \mathtt{M}) \leftarrow \mathrm{e}(\mathtt{X}, \mathtt{L}), \mathbf{not}(\mathrm{e}(\mathtt{X}, \mathtt{M}));$$
$$2: \mathrm{e}(\mathtt{X}, \mathtt{X}.\mathtt{L}); \qquad 3: \mathrm{e}(\mathtt{X}, \mathtt{Y}.\mathtt{L}) \leftarrow \mathrm{e}(\mathtt{X}, \mathtt{L});$$

Дерево SLDNF-резолютивных вычислений этой программы для $?\mathrm{p}(\mathtt{X}, \mathbf{a.b.nil}, \mathbf{b.c.nil})$ и стандартной стратегии вычисления:



Математическая логика и логическое программирование

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

Блок 48

Модальные логики

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

ВМК МГУ, 2023/2024, осенний семестр

Модальности

Рассмотрим такие высказывания:

1: **Зима близко**

2: **Зима всегда близко**

3: **Зима бывает близко**

Если отбросить всё лишнее, то
для высказывания *1* справедлива одна из двух оценок:
правда, если зима действительно близко, и
неправда, если это не так

Смысл высказываний *2* и *3* *тесно связан* со смыслом *1*:
если $x =$ «зима близко», то

1: x *2:* всегда x *3:* иногда бывает x

Модальности

Рассмотрим такие высказывания:

1: **Зима близко**

2: **Зима всегда близко**

3: **Зима бывает близко**

Слова, используемые как уточнение истинностной оценки высказывания, называются модальностями (*лат. modus — мера*)

«Всегда» и «иногда» — это темпоральные модальности (модальности времени) (*лат. tempus — время*)

Можно сказать, что «всегда» и «иногда» — это $\forall$ и $\exists$, но тогда:

- Какими предметами задаётся время, и каковы свойства этих предметов?
- Как устроить анализ смысла высказываний в интерпретациях с такими предметами?

Модальности

Рассмотрим такие высказывания:

1: **Зима близко**

2: **Я знаю, что зима близко**

3: **Я допускаю возможность того, что зима близко**

Смысл высказываний *2* и *3* *тесно связан* со смыслом *1*:
если $x =$ «зима близко», то

1: x *2:* известно, что x *3:* допустимо x

«Известно» и «допустимо» — это эпистемические модальности
(модальности знания) (*др.-греч.* ἐπιστήμη — *знание*)

Если «известно» (*всеобщее знание*)
и «допустимо» (*частное, некоторое знание*) трактовать как $\forall$ и $\exists$,
то на какие «предметы знаний» указывают эти кванторы?

Модальности

Рассмотрим такие высказывания:

1: **Зима близко**

2: **Зима должна быть близко**

3: **Зима имеет право быть близко**

Смысл высказываний *2* и *3* *тесно связан* со смыслом *1*:
если $x =$ «зима близко», то

1: x *2:* должно быть x *3:* имеет право быть x

«Должен» и «имеет право» — это деонтические модальности
(модальности долга) (*др.-греч.* δέον — *должное*)

Как связан смысл фраз «это так» и «это должно быть так»?

В *рациональных* высказываниях используется великое множество разных модальностей, и точно описать их смысл в рамках классической логики — очень непростая задача

Модальности

Часто (*хотя и не всегда*) в высказываниях используются модальности двух двойственных видов:

Модальность необходимого	**Модальность возможного**
необходимо	возможно
обязательно	не исключено
всегда	иногда
должен	имеет право
знает	предполагает
доказуемо	непротиворечиво
$\Box$	$\Diamond$

Хотелось бы иметь единообразный способ определения смысла модальностей $\Box$ и $\Diamond$ — и такой способ используется в

модальных логиках

Модальные логики: синтаксис

Синтаксис формул модальной логики высказываний
над множеством пропозициональных переменных Var:

$$\varphi \ ::= \ \mathtt{x} \ \mid \ (\neg\varphi) \ \mid \ (\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\Box\varphi) \ \mid \ (\Diamond\varphi) \ ,$$

где φ — формула и $\mathtt{x} \in \mathrm{Var}$

Это синтаксис формул логики высказываний,
расширенный возможностью расстановки модальностей
необходимого и возможного над любыми (под)формулами

Приоритет операций: $\neg$, $\Box$ и $\Diamond$; затем $\&$; затем $\vee$; затем $\to$

Пример формулы: $\Diamond\mathtt{x} \,\&\, \Box\neg\Diamond(\mathtt{x} \vee \mathtt{y})$
«возможно $\mathtt{x}$*, и при этом необходимо невозможно, что* $\mathtt{x}$ *или* $\mathtt{y}$*»*

В этой формуле не говорится,
что означают «необходимость» и «возможность»:
точный смысл $\Box$ и $\Diamond$ — это часть семантики формулы

Модальные логики: семантика Крипке

Описание семантики начнём с примера: **Зима всегда близко**

Представим себе любое строгое определение зимы
и того, в каких случаях зима считается близкой

Если остановить время и задаться вопросом, близко зима или нет,
то ответ на этот вопрос (**да** или **нет**) можно представить как
значение пропозициональной переменной x

Время течёт, и мир меняется, как и значение x:

$$\underset{x}{\text{сейчас}} \longrightarrow \underset{x}{\text{через минуту}} \longrightarrow \underset{x}{\text{через месяц}} \longrightarrow \underset{\neg x}{\text{через полгода}}$$

Ответ на вопрос «$\Box x$?» в так меняющемся мире — **нет**, не всегда x

Изменение мира можно представить себе и по-другому:
существует много взаимосвязанных миров с разными значениями x,
и в семантике $\Box$ комбинируются значения x многих миров

Модальные логики: семантика Крипке

Модель Крипке (над переменными Var) — это система $(W, \mapsto, L)$, где:

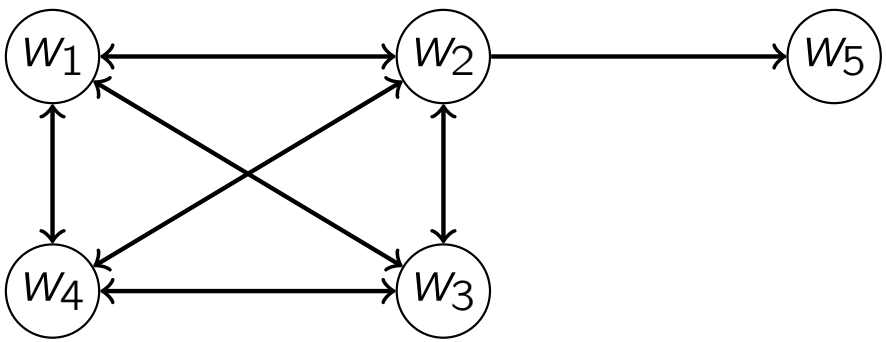
- W — непустое множество миров (или, по-другому, — состояний)
- $\mapsto \subseteq W \times W$ — отношение переходов между мирами
  - Если $w \mapsto w'$, то w' — мир, альтернативный для w (w-альтернатива)
- $L : W \to 2^{\mathrm{Var}}$ — оценка переменных для каждого мира
  - «$\mathtt{x} \in L(w)$» = «переменная $\mathtt{x}$ истинна в мире w»

Модель Крипке — это интерпретация для формул модальной логики

Шкала Крипке (*Kripke frame*),
на которой основывается модель $(W, \mapsto, L)$, — это пара $(W, \mapsto)$

Модальные логики: семантика Крипке

Например, $(\mathrm{Var} = \{a\})$



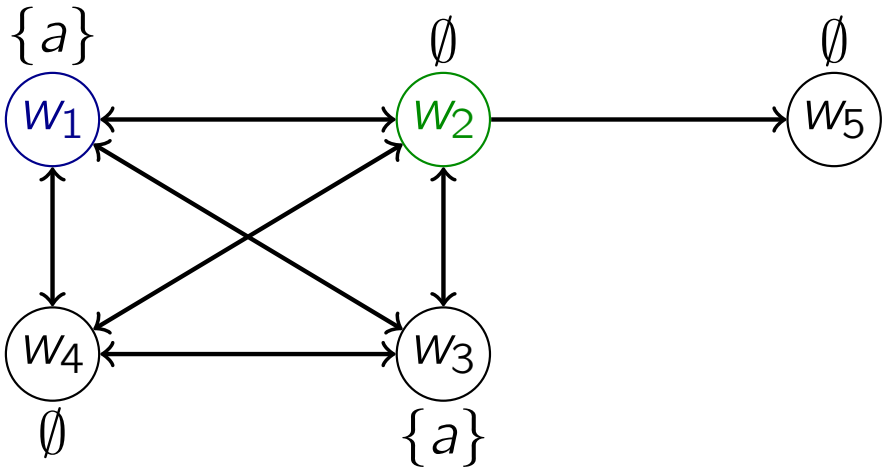
— это шкала Крипке

Модальные логики: семантика Крипке

Отношение выполнимости $\models$ для модели $\mathcal{I} = (W, \mapsto, L)$ и мира $w \in W$ определяется так:

- $\mathcal{I}, w \models \mathrm{x} \quad \Leftrightarrow \quad \mathrm{x} \in L(w)$ $\qquad (\mathrm{x} \in \mathrm{Var})$
- $\mathcal{I}, w \models \varphi \,\&\, \psi \quad \Leftrightarrow \quad \mathcal{I}, w \models \varphi$ и $\mathcal{I}, w \models \psi$
- $\mathcal{I}, w \models \varphi \vee \psi \quad \Leftrightarrow \quad \mathcal{I}, w \models \varphi$ или $\mathcal{I}, w \models \psi$

Например, $\qquad (\mathrm{Var} = \{a\})$



— это модель Крипке $(\mathcal{I})$

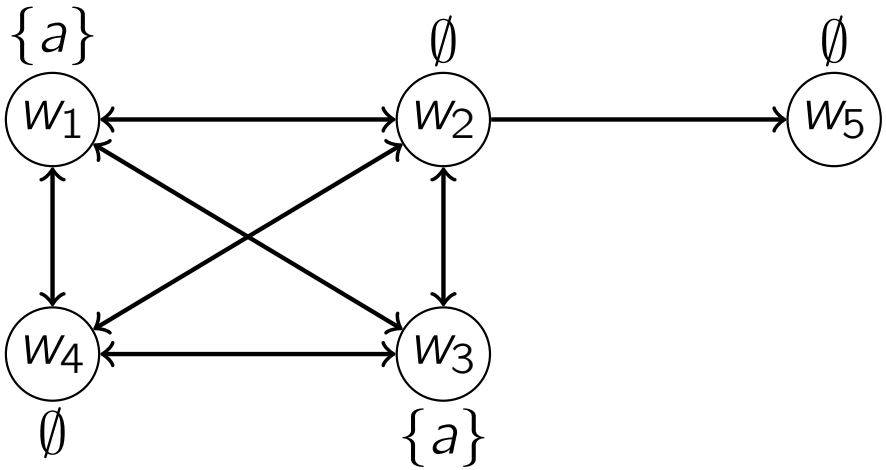
$$\mathcal{I}, w_1 \models a, \qquad \mathcal{I}, w_2 \not\models a$$

Модальные логики: семантика Крипке

Отношение выполнимости $\models$ для модели $\mathcal{I} = (W, \mapsto, L)$ и мира $w \in W$ определяется так:

- $\mathcal{I}, w \models \varphi \to \psi \quad \Leftrightarrow \quad \mathcal{I}, w \not\models \varphi$ или $\mathcal{I}, w \models \psi$
- $\mathcal{I}, w \models \neg\varphi \quad \Leftrightarrow \quad \mathcal{I}, w \not\models \varphi$

Например, $(\mathrm{Var} = \{a\})$



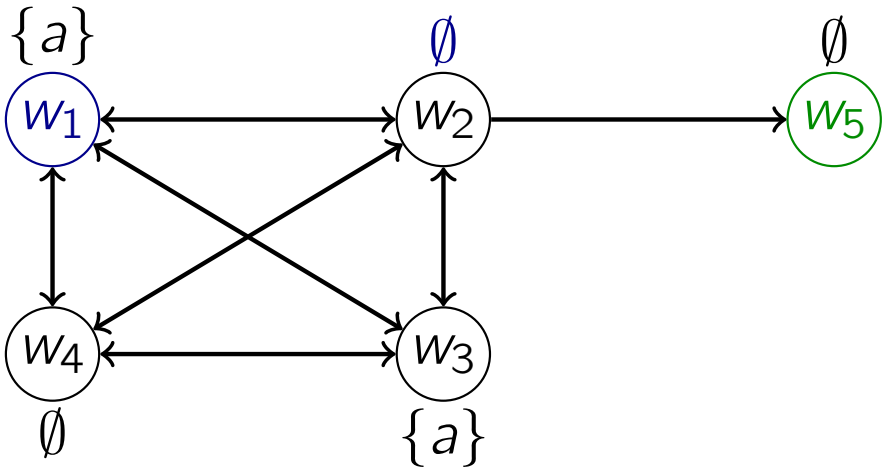
— это модель Крипке $(\mathcal{I})$

Модальные логики: семантика Крипке

Отношение выполнимости $\models$ для модели $\mathcal{I} = (W, \mapsto, L)$ и мира $w \in W$ определяется так:

- $\mathcal{I}, w \models \Box\varphi \quad \Leftrightarrow$
  для любой w-альтернативы w' верно $\mathcal{I}, w' \models \varphi$

Например, $(\mathrm{Var} = \{a\})$



— это модель Крипке $(\mathcal{I})$

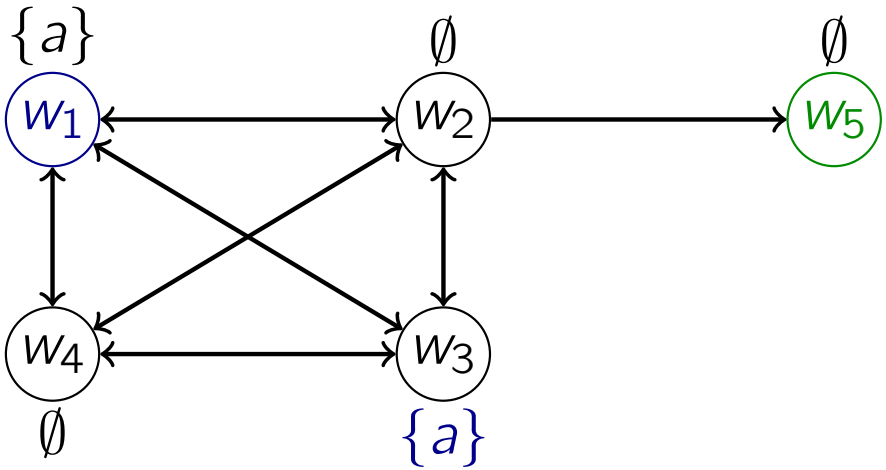
$\mathcal{I}, w_1 \not\models \Box a, \qquad \mathcal{I}, w_5 \models \Box a$

Модальные логики: семантика Крипке

Отношение выполнимости $\models$ для модели $\mathcal{I} = (W, \mapsto, L)$ и мира $w \in W$ определяется так:

- $\mathcal{I}, w \models \Diamond\varphi \quad \Leftrightarrow$
  существует w-альтернатива w',
  такая что верно $\mathcal{I}, w' \models \varphi$

Например, $(\mathrm{Var} = \{a\})$

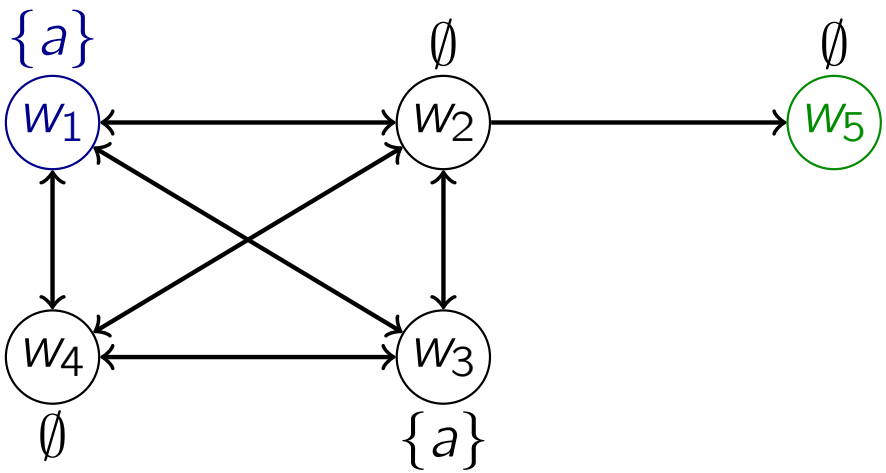


— это модель Крипке $(\mathcal{I})$

$\mathcal{I}, w_1 \models \Diamond a, \qquad \mathcal{I}, w_5 \not\models \Diamond a$

Модальные логики: семантика Крипке

Например, $(\mathrm{Var} = \{a\})$

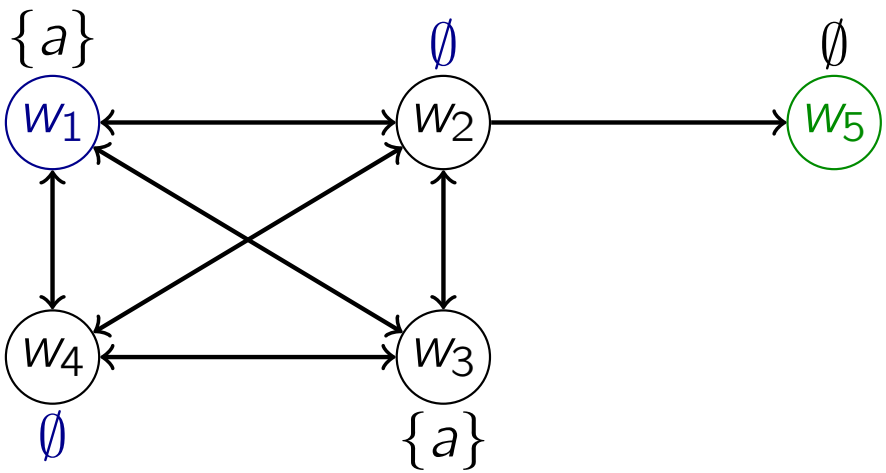


— это модель Крипке $(\mathcal{I})$

$\mathcal{I}, w_1 \models \Box\Diamond a, \qquad \mathcal{I}, w_5 \models \Box\Diamond a$

Модальные логики: семантика Крипке

Например, $(\mathrm{Var} = \{a\})$



— это модель Крипке $(\mathcal{I})$

$\mathcal{I}, w_1 \not\models \Diamond\Box a,$ $\mathcal{I}, w_2 \models \Diamond\Box a$, $\mathcal{I}, w_5 \not\models \Diamond\Box a$

Модальные логики: семантика Крипке

Пусть $\mathcal{F}$ — шкала Крипке, $\mathcal{I}$ — модель Крипке, и φ, ψ — формулы модальной логики

Тогда

- формула φ истинна в модели $\mathcal{I}$ ($\mathcal{I} \models \varphi$), если для любого мира w модели $\mathcal{I}$ верно $\mathcal{I}, w \models \varphi$
- формула φ истинна на шкале $\mathcal{F}$ ($\mathcal{F} \models \varphi$), если для любой модели Крипке $\mathcal{J}$, основанной на $\mathcal{F}$, верно $\mathcal{J} \models \varphi$
- формула φ общезначима ($\models \varphi$), если для любой шкалы $\mathcal{F}$ верно $\mathcal{F} \models \varphi$
- формулы φ и ψ равносильны ($\varphi \sim \psi$), если $\models \varphi \leftrightarrow \psi$

Законы модальных логик

Модальная логика, как и любая другая, имеет свои законы и свойства, верные независимо от точного смысла модальностей — например:

Утверждение. **Для любой формулы φ верно:** $\Box\varphi \sim \neg\Diamond\neg\varphi$

Доказательство. «В любой альтернативе выполняется φ» $\sim$ «Не существует альтернативы, в которой φ не выполняется» ▼

Утверждение. **Для любой формулы φ верно: если $\models \varphi$, то $\models \Box\varphi$**

Доказательство. Если φ выполняется в любом мире любой модели, то φ выполняется и в любой альтернативе любого мира любой модели ▼

Законы модальных логик

Модальная логика, как и любая другая, имеет свои законы и свойства, верные независимо от точного смысла модальностей — например:

Утверждение. Для любых формул φ, ψ верно:
$\models \Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$

Доказательство.

Предположим от противного, что $\not\models \Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$

Тогда существуют модель $\mathcal{I} = (W, \mapsto, L)$ и мир w, такие что
$\mathcal{I}, w \not\models \Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$

Тогда по семантике $\to$:

- $\mathcal{I}, w \models \Box(\varphi \to \psi)$, а значит,
  для каждой w-альтернативы w' верно $\mathcal{I}, w' \models \varphi \to \psi$
- $\mathcal{I}, w \models \Box\varphi$, а значит, для каждой w-альтернативы w'
  верно $\mathcal{I}, w' \models \varphi$ — и, по семантике $\to$, $\mathcal{I}, w' \models \psi$
- $\mathcal{I}, w \not\models \Box\psi$, а значит, существует w-альтернатива w',
  такая что $\mathcal{I}, w' \not\models \psi$, что *противоречит* предыдущему пункту ▼

Математическая логика и логическое программирование

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

Блок 49

Эпистемические логики

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

ВМК МГУ, 2023/2024, осенний семестр

Эпистемическая логика (логика знаний) — это разновидность модальной логики, в которой модальность $\Box$ означает «я знаю», а $\Diamond$ — «я допускаю»

Так как эпистемическая логика является модальной, то в ней справедливы все законы модальной логики

Но смыслом модальностей могут определяться и другие законы

Если «я» — **идеальный познающий субъект**, то совокупность моих знаний и допущений должна подчиняться, помимо общих законов модальных логик, *как минимум* таким законам:

- мои знания верны
  $\Box\varphi \to \varphi$ (закон адекватности знания)
- мне известно, что именно я знаю
  $\Box\varphi \to \Box\Box\varphi$ (закон позитивной интроспекции)
- мне известно, что именно я **не** знаю
  $\neg\Box\varphi \to \Box\neg\Box\varphi$ (закон негативной интроспекции)

Двуместное отношение $\mapsto$ на множестве W

- рефлексивно, если для любого элемента w множества W верно $w \mapsto w$
- симметрично, если для любых элементов w_1, w_2 множества W справедлива импликация

$$w_1 \mapsto w_2 \quad \Rightarrow \quad w_2 \mapsto w_1$$

- транзитивно, если для любых элементов w_1, w_2, w_3 множества W справедлива импликация

$$w_1 \mapsto w_2 \mapsto w_3 \quad \Rightarrow \quad w_1 \mapsto w_3$$

Утверждение. **Для любой шкалы Крипке $\mathcal{F} = (W, \mapsto)$ верно:**

$\mathcal{F} \models \Box\varphi \to \varphi$ верно для любой формулы φ

$\Leftrightarrow$

отношение $\mapsto$ рефлексивно

Доказательство.

($\Rightarrow$) Пусть отношение $\mapsto$ нерефлексивно

Тогда существует мир w, такой что $w \not\mapsto w$

Рассмотрим переменную p и модель Крипке $\mathcal{I} = (W, \mapsto, L)$, такие что:

- $p \notin L(w)$
- Для любой w-альтернативы w' верно $p \in L(w')$

По выбору w, модель $\mathcal{I}$ задана корректно

При этом $\mathcal{I}, w \models \Box p$ и $\mathcal{I}, w \not\models p$, а значит, и $\mathcal{I}, w \not\models \Box p \to p$

Следовательно, $\mathcal{F} \not\models \Box p \to p$

Утверждение. **Для любой шкалы Крипке $\mathcal{F} = (W, \mapsto)$ верно:**

$\mathcal{F} \models \Box\varphi \to \varphi$ верно для любой формулы φ

$\Leftrightarrow$

отношение $\mapsto$ рефлексивно

Доказательство.

($\Leftarrow$) Пусть отношение $\mapsto$ рефлексивно

Предположим от противного, что существуют формула φ, модель Крипке $\mathcal{I} = (W, \mapsto, L)$ и её мир w, такие что $\mathcal{I}, w \not\models \Box\varphi \to \varphi$

По семантике $\to$ и $\Box$:

1. $\mathcal{I}, w \not\models \varphi$
2. Для любой w-альтернативы w' верно $\mathcal{I}, w' \models \varphi$

По рефлексивности $\mapsto$, мир w является w-альтернативой

Значит, $\mathcal{I}, w \models \varphi$, что *противоречит* первому пункту ▼

Утверждение. **Для любой шкалы Крипке $\mathcal{F} = (W, \mapsto)$ верно:**

$\mathcal{F} \models \Box\varphi \to \Box\Box\varphi$ **верно для любой формулы** φ

$\Leftrightarrow$

отношение $\mapsto$ **транзитивно**

Утверждение. **Для любой шкалы Крипке $\mathcal{F} = (W, \mapsto)$ с рефлексивным и транзитивным отношением** $\mapsto$ **верно:**

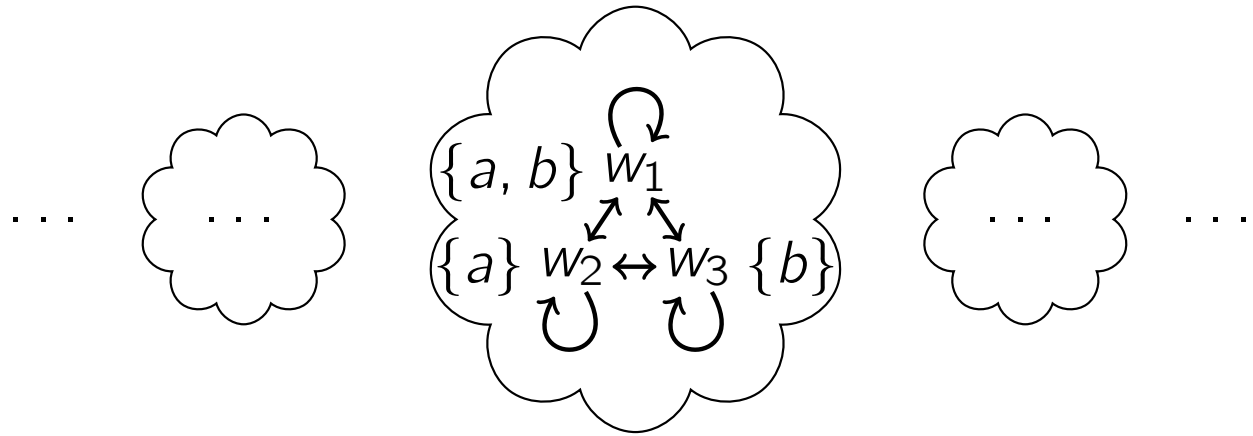
$\mathcal{F} \models \neg\Box\varphi \to \Box\neg\Box\varphi$ **верно для любой формулы** φ

$\Leftrightarrow$

отношение $\mapsto$ **симметрично**

Можете попробовать самостоятельно доказать эти утверждения по аналогии с утверждением про рефлексивность

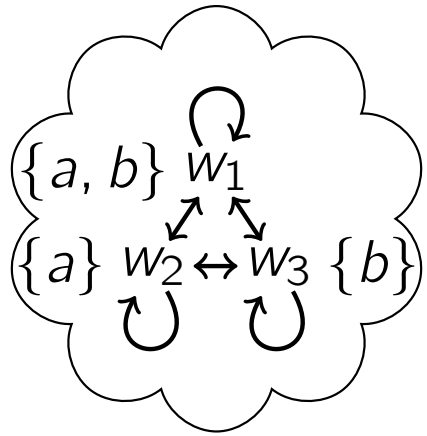
Следствие. **Модель Крипке идеального познающего субъекта — это модель, отношение переходов которой является отношением эквивалентности**



Пояснение

Пусть w_1 — мир достоверных фактов

Тогда класс эквивалентности w_1 состоит из всех миров, устройство которых не противоречит информации, которой располагает познающий субъект



Пояснение

- a — это факт, ...

$$\mathcal{I}, w_1 \models a$$

- ..., но моих знаний недостаточно, чтобы это утверждать, ...

$$\mathcal{I}, w_1 \not\models \Box a$$

- ..., но и опровергнуть a я тоже не могу,

$$\mathcal{I}, w_1 \models \Diamond a$$

- а что я точно знаю, так это то, что если не a, то обязательно b

$$\mathcal{I}, w_1 \models \Box(\neg a \to b)$$

В более широкой и «полезной» постановке задачи познающий субъект

- может изменять мир согласно своим скромным возможностям
- может взаимодействовать с другими такими же субъектами, обмениваясь с ними знаниями
- пытается достичь некоторой цели, кооперируясь или конкурируя с другими субъектами

Таких взаимодействующих субъектов принято называть агентами, а совокупность всех агентов с описанием их возможностей и целей — мультиагентной системой

Каждому агенту a такой системы присваивается своя эпистемическая модальность $\Box_a$: «агент a знает, что ...»

Иногда рассматриваются и групповые модальности — например, $\Box_\forall$: «все агенты знают, что ...»

Пример: задача о трёх мудрецах

Король призвал трёх мудрецов,
показал им три чёрные шапки и две белые, завязал глаза,
надел на мудрецов чёрные шапки, спрятал белые и развязал глаза

«Из пяти шапок, что я показал, три надеты на вас», — сказал король
«Знаете ли вы, какая на вас шапка?» — спросил король

«Нет, не знаю», хором ответили мудрецы

«Знаете ли вы, какая на вас шапка?» — повторил король

«Нет, не знаю», хором ответили мудрецы

«Знаете ли вы, какая на вас шапка?» — ещё раз повторил король

«**Да, чёрная**», хором ответили мудрецы

Как может выглядеть ход рассуждений мудрецов
в терминах эпистемической логики для мультиагентной системы?

Математическая логика и логическое программирование

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

Блок 50

Темпоральные логики

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

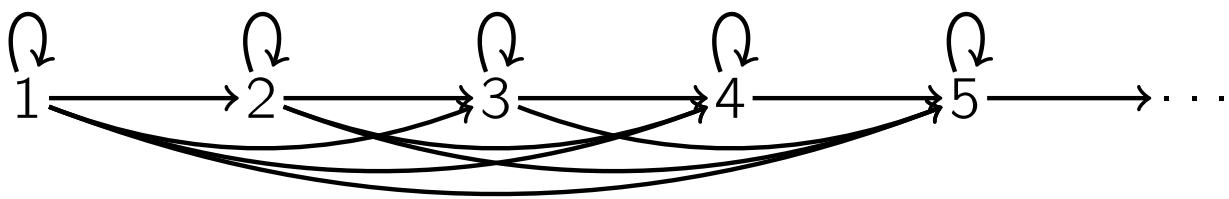
ВМК МГУ, 2023/2024, осенний семестр

Темпоральные логики

Темпоральная логика (логика времени) — это разновидность модальной логики, в которой $\Box$ и $\Diamond$ отвечают необходимости и возможности выполнимости формулы в условиях течения времени, обычно «всегда [в будущем]» и «иногда [в будущем]»

Шкалой Крипке темпоральной логики описывается течение времени: миры — это моменты времени, а отношением достижимости миров описывается порядок смены моментов времени

Пример: шкала дискретного линейного отсчёта времени



Пропозициональные переменные формул темпоральных логик — это атомарные высказывания, истинность которых может изменяться с течением времени

Темпоральные логики

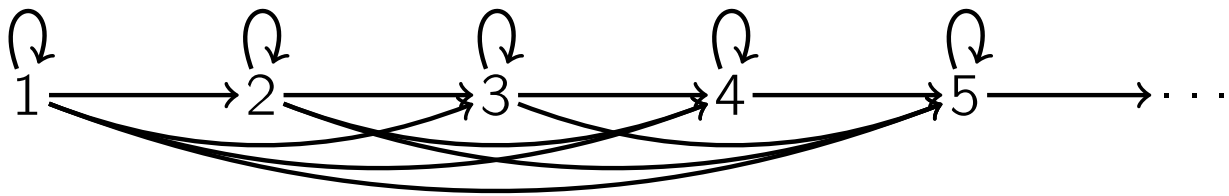
Течение времени может истолковываться по-разному, и в зависимости от истолкования (*то есть точного вида рассматриваемых шкал*) могут получаться разные темпоральные логики, например:

- Логика линейного времени
  (**LTL**, **L**inear **T**emporal **L**ogic)
  - время дискретно линейно течёт вперёд
  - формула — это свойство линейного развития событий

- Логика деревьев вычислений
  (**CTL**, **C**omputation **T**ree **L**ogic)
  - время — это частично упорядоченное множество, которым описываются все варианты развития событий
  - формула — это высказывание о возможности и невозможности заданного развития событий с учётом всех вариантов

Логика линейного времени (LTL)

Шкала LTL — это естественно упорядоченный натуральный ряд (моментов времени):



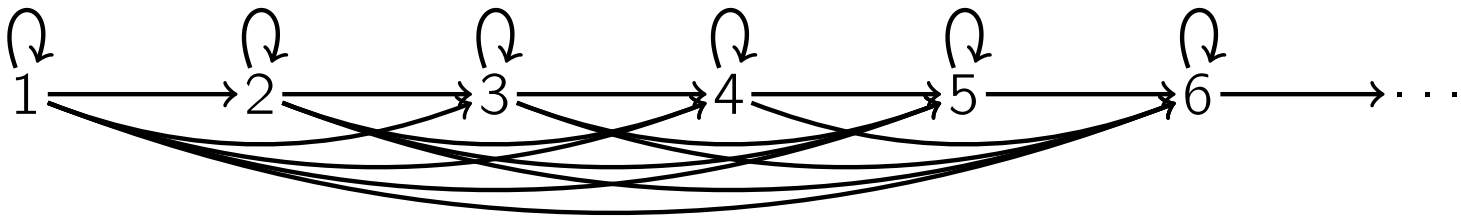
Интерпретация LTL — это модель Крипке, основанная на шкале LTL

Модальности $\Box$ и $\Diamond$ в LTL обычно обозначаются символами **G** (**G**lobally) и **F** (in **F**uture)

К ним могут добавляться и другие модальности, описывающие те или иные взаимосвязи между высказываниями в неуклонно текущем времени

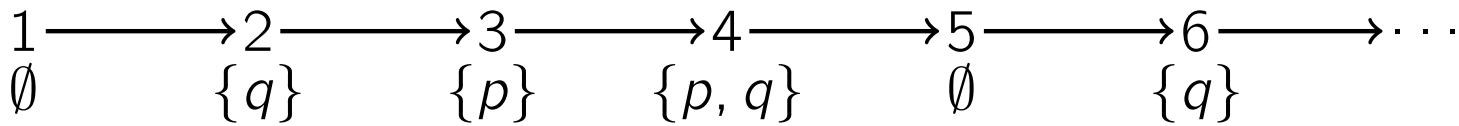
Логика линейного времени (LTL)

Пример:



Логика линейного времени (LTL)

Пример: рассмотрим такую интерпретацию $\mathcal{I}$ LTL
с оценкой атомарных высказываний, повторяющейся с периодом 4:

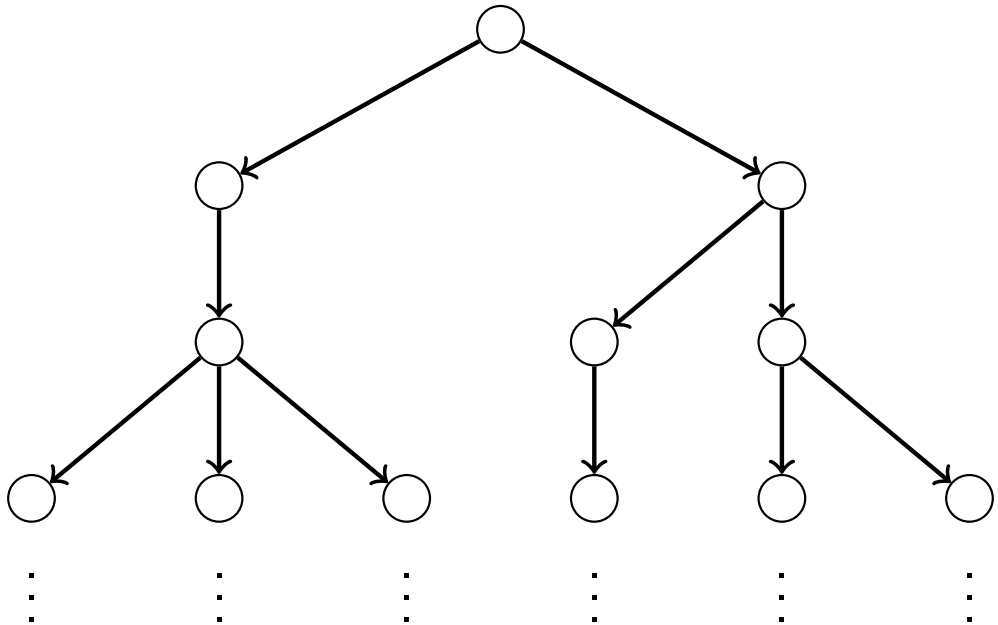


Справедливы следующие соотношения:

- в момент времени 1 неверно p и верно $p \to q$
$$\mathcal{I}, 1 \not\models p, \qquad \mathcal{I}, 1 \models p \to q$$
- p иногда бывает верным, но не всегда
$$\mathcal{I}, 1 \models \mathbf{F}p, \qquad \mathcal{I}, 1 \not\models \mathbf{G}p$$
- p бесконечно часто бывает верным,
но нет такого момента, начиная с которого p всегда верно
$$\mathcal{I}, 1 \models \mathbf{GF}p, \qquad \mathcal{I}, 1 \not\models \mathbf{FG}p$$

Логика деревьев вычислений (CTL)

Течение времени в CTL описывается ориентированным деревом, содержащим только бесконечные ветви:



Шкала CTL — это шкала Крипке,
являющаяся рефлексивно-транзитивным замыканием такого дерева:
миры — это вершины дерева,
а отношение переходов — это отношение достижимости миров в дереве

Интерпретация CTL — это модель Крипке, основанная на Шкале CTL

Логика деревьев вычислений (CTL)

В логике деревьев вычислений модальности $\Box$, $\Diamond$ записываются как **AG** (for **A**ll paths **G**) и **EF** (**E**xists path such that **F**)

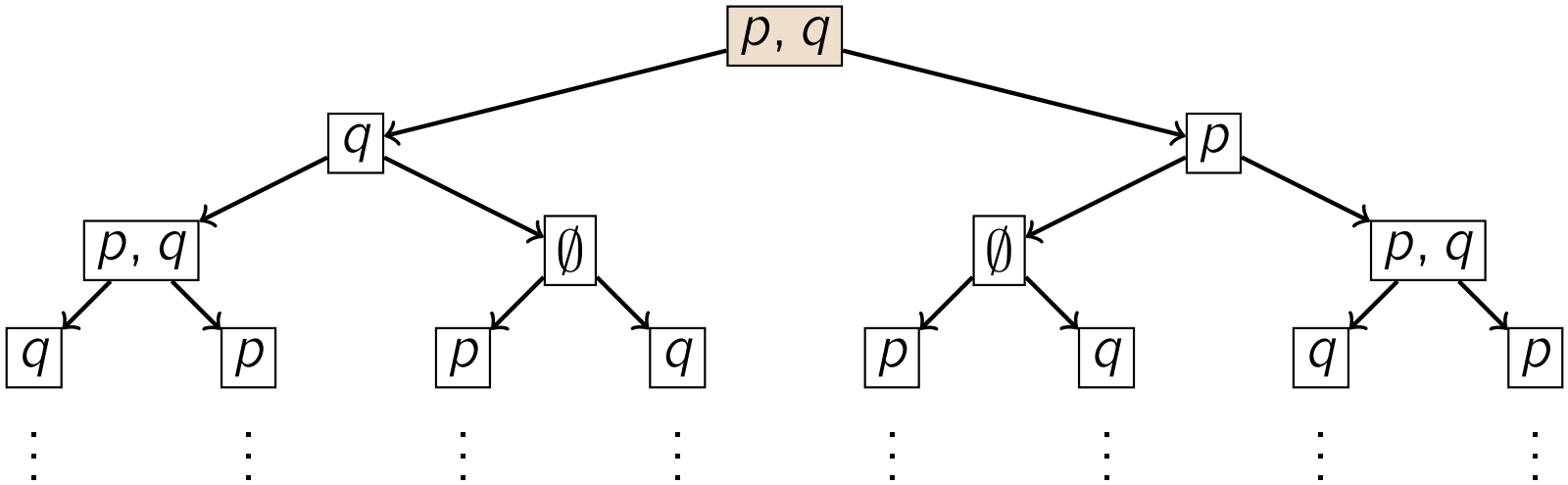
CTL содержит и другие модальности — например, **EG** и **AF**

Смысл этих модальностей определяется так:

- $\mathcal{I}, v \models \mathbf{EG}\varphi \Leftrightarrow$ существует ветвь дерева, исходящая из v и такая что для каждой вершины w этой ветви верно $\mathcal{I}, w \models \varphi$
- $\mathcal{I}, v \models \mathbf{AF}\varphi \Leftrightarrow$ в каждой ветви дерева, исходящей из v, существует вершина w, такая что $\mathcal{I}, w \models \varphi$

Логика деревьев вычислений (CTL)

Пример: рассмотрим такую интерпретацию $\mathcal{I}$ CTL
(при переходе влево изменяется значение p, вправо — значение q):

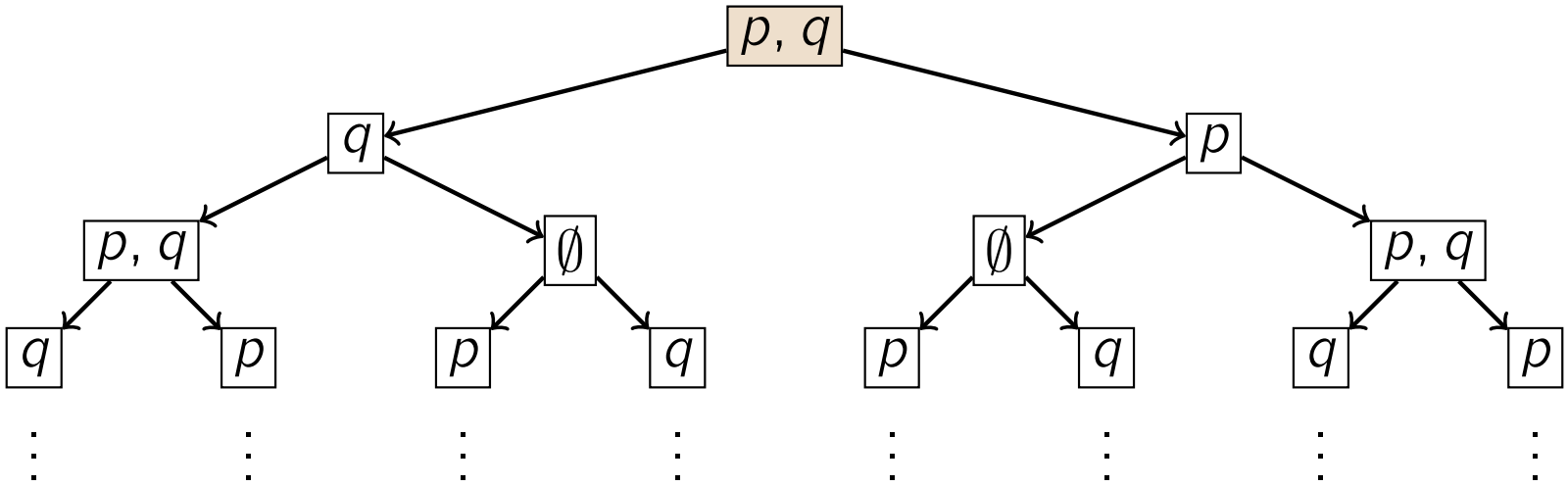


Справедливы следующие соотношения:

- в начале развития событий верно p,
  но события могут развиваться так, что p станет неверным
  $$\mathcal{I}, \square \models p, \qquad \mathcal{I}, \square \models \mathbf{EF}\neg p$$
- события могут развиваться так, чтобы p всегда оставалось верным,
  но могут развиваться и по-другому
  $$\mathcal{I}, \square \models \mathbf{EG}p, \qquad \mathcal{I}, \square \not\models \mathbf{AG}p$$

Логика деревьев вычислений (CTL)

Пример: рассмотрим такую интерпретацию $\mathcal{I}$ CTL
(при переходе влево изменяется значение p, вправо — значение q):



Справедливы следующие соотношения:

- как бы ни развивались события до сих пор, есть способ в дальнейшем сделать p и q одновременно верными
$$\mathcal{I}, \square \models \mathbf{AGEF}(p \,\&\, q)$$
- утверждение «после возникновения события p рано или поздно неотвратимо возникает событие q» неверно
$$\mathcal{I}, \square \not\models \mathbf{AG}(p \to \mathbf{AF}q)$$

Математическая логика и логическое программирование

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

Блок 51

Интуиционистская логика

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Интуиционизм — это философское и математическое течение, возникшее в начале XX века как ответ на чересчур широкое и вольное использование формальных логических методов и «нереальных» бесконечных объектов в математических рассуждениях

Сжато и упрощённо можно представить основную идею интуиционистского взгляда на математику так:

- Чтобы утверждение было признано истинным, требуется «убедительное» «конструктивное» доказательство этого утверждения
- В частности, если утверждается существование некоего конструкта, то доказательством должен предоставляться способ его «наглядного» «конструктивного» построения
- Все остальные доказательства — это и не доказательства вовсе, а просто словоблудие

Вступление

Как и ряд других «радикальных» течений, существовавших в начале XX века, интуиционизм

- не сохранился в современной математике «в чистом виде»,
- но внёс свой вклад в устройство математики в целом
- и в упрощённом переосмысленном виде сохранился в виде особого логического языка, основанного на
  - необычном (неклассическом) спектре логических законов
  - и соответствующем необычном понимании истинности утверждений

Этот язык и известен сейчас под названием «интуиционистская логика»

Вступление

Пример того, что современные математики считают корректным доказательством, а интуиционисты считали словоблудием

Теорема (?). Существуют иррациональные чи́сла α и β, такие что α^β — рациональное число

Доказательство (?)

Если $\sqrt{2}^{\sqrt{2}}$ — рациональное число, то $\alpha = \beta = \sqrt{2}$

Иначе $\alpha = \sqrt{2}^{\sqrt{2}}$ и $\beta = \sqrt{2}$ (т.к. $\alpha^\beta = (\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\cdot\sqrt{2}} = 2$) ▼

Эти слова звучат достаточно убедительно и склоняют к тому, чтобы поверить, что заявленные чи́сла α и β действительно существуют

Но чему же всё-таки равны α и β?

По мнению интуиционистов, пока не показано, как можно представить себе эти чи́сла при помощи конструктивных умозаключений, утверждение об их существовании — это всего лишь не вполне аргументированная попытка убедить собеседника в своей правоте, которая не может считаться абсолютной истиной

Семантика Колмогорова-Брауэра-Гейтинга

В интуиционистской логике высказываний

формулами являются формулы логики высказываний:

$$\varphi \ ::= \ \mathtt{x} \ \mid \ (\varphi \,\&\, \varphi) \ \mid \ (\varphi \vee \varphi) \ \mid \ (\varphi \to \varphi) \ \mid \ (\neg\varphi),$$

где φ — формула, $\mathtt{x} \in \mathrm{Var}$ и Var — множество пропозициональных переменных

Попробуем переосмыслить эти формулы так:

- Каждая переменная отвечает некоторой атомарной *задаче*
- Истинность переменной означает, что в нашем распоряжении имеется решение соответствующей задачи
  - (И не просто утверждение о том, что решение существует, а именно само решение — например, явное описание алгоритма)
- Составные формулы отвечают составным задачам, конструирующимся из более простых при помощи способов, задающихся связками
- Истинность составных формул означает, что в нашем распоряжении имеется решение соответствующей составной задачи

Семантика Колмогорова-Брауэра-Гейтинга

Составная задача	*Для решения этой составной задачи требуется ...*
$\varphi \,\&\, \psi$	предъявить решение каждой из подзадач φ, ψ
$\varphi \vee \psi$	выбрать одну из подзадач φ, ψ и предъявить её решение
$\varphi \to \psi$	предъявить *сведéние* задачи ψ к задаче φ, то есть решение задачи ψ в предположении о том, что в нашем распоряжении имеется решение задачи φ
$\neg\varphi$	доказать, что задача φ не имеет решения

Формула считается законом интуиционистской логики (*общезначимой*), если она отвечает составной задаче, имеющей решение независимо от наличия решений атомарных задач

Семантика Колмогорова-Брауэра-Гейтинга

Примеры законов интуиционистской логики

- $\varphi \to \varphi$: «каждую задачу можно тривиально свести к ней самой»
- $(\varphi \to \psi)\,\&(\psi \to \chi) \to (\varphi \to \chi)$:
  «чтобы свести задачу χ к задаче φ, достаточно найти задачу ψ, такую что к ней сводится χ и она сводится к φ»
- $\varphi \to \neg\neg\varphi$:
  «чтобы доказать, что невозможно предъявить доказательство того, что задача не имеет решения,
  достаточно предъявить решение этой задачи»
- $(\neg\varphi \vee \neg\psi) \to \neg(\varphi \,\&\, \psi)$:
  «чтобы доказать, что невозможно одновременно решить обе задачи φ, ψ, достаточно выбрать одну из этих задач и доказать, что она не имеет решения»

Семантика Колмогорова-Брауэра-Гейтинга

Примеры формул, не являющихся законами интуиционистской логики

Вообще говоря, не является верным следующее:

- $\neg\neg\varphi \to \varphi$:
  «чтобы получить решение задачи, достаточно доказать,
  что невозможно предъявить доказательство того,
  что эта задача не имеет решения»

- $\varphi \vee \neg\varphi$:
  «для любой задачи мы располагаем либо решением,
  либо доказательством того, что решение не существует»

- $\neg(\varphi \,\&\, \psi) \to (\neg\varphi \vee \neg\psi)$:
  «чтобы суметь выбрать одну из задач φ, ψ
  и доказать, что она не имеет решения, достаточно доказать,
  что невозможно решить φ и ψ одновременно»

Семантика Колмогорова-Брауэра-Гейтинга

Чтобы все предложенные примеры и построения не оказались таким же словоблудием, как и то, ради устранения чего они придумывались, следует строго и непротиворечиво определить семантику формул так, чтобы она соответствовала общим представлениям о наличии и отсутствии решений разнообразных задач

Чтобы лучше понять такую семантику, представим себе то, как может учиться решать атомарные задачи идеальный математик:

- В каждый момент времени он
  - умеет решать некоторые атомарные задачи,
  - не умеет решать остальные атомарные задачи
  - и точно знает, что он умеет решать, а что нет
- Его умения изменяются со временем:
  - он может научиться решать ту задачу, которую не умел решать,
  - но может и не научиться,
  - и совершенно точно он никогда не разучится решать те задачи, которые научился решать

Интуиционистская модель Крипке

Пояснения про идеального математика наводят на мысль, что семантика формул может быть основана на понятиях модальной логики

Это действительно так, и это один из наиболее популярных способов задания семантики интуиционистских формул

Шкалу Крипке $\mathcal{F} = (W, \preceq)$ будем называть интуиционистской, если отношение $\preceq$ является нестрогим частичным порядком:

- рефлексивно, транзитивно и
- антисимметрично: для любых миров w_1, w_2 верна импликация

$$w_1 \preceq w_2 \text{ и } w_2 \preceq w_1 \quad \Rightarrow \quad w_1 = w_2$$

Интуиционистская интерпретация — это модель Крипке $(W, \preceq, L)$, для которой верно следующее:

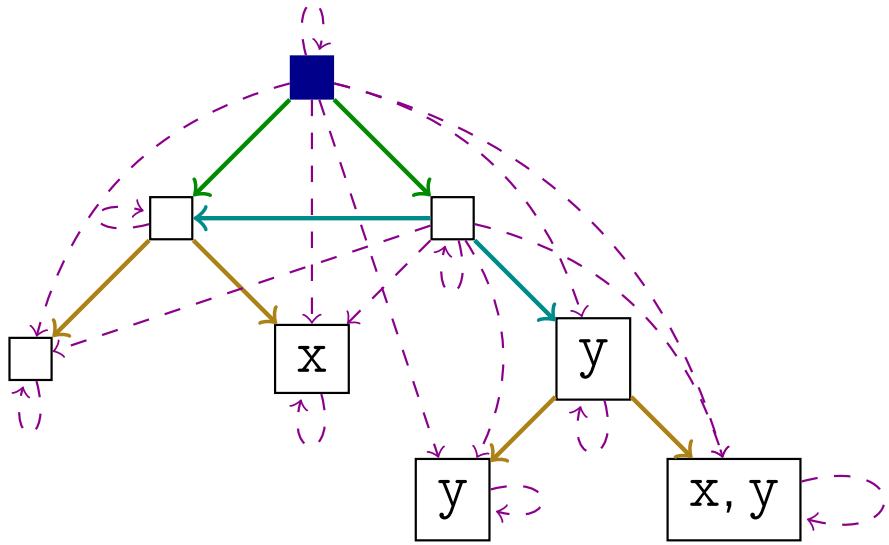
- $(W, \preceq)$ — интуиционистская шкала Крипке
- оценка L монотонна: для любых миров w_1, w_2 и любой переменной x верна импликация

$$\mathrm{x} \in L(w_1) \text{ и } w_1 \preceq w_2 \quad \Rightarrow \quad \mathrm{x} \in L(w_2)$$

Интуиционистская модель Крипке

Пример интуиционистской интерпретации над переменными

- x: умею программировать в логической парадигме
- y: умею доказывать теорему Чёрча



Начало семестра

Пошёл на лекцию по теореме Чёрча либо прогулял её

Проспал эту лекцию либо слушал внимательно

Независимо от лекции, участвовал в семинарах либо нет

Плюс рефлексивность и транзитивность (такие переходы в остальных примерах будут опускаться)

Интуиционистское отношение выполнимости

Выполнимость формулы φ интуиционистской логики в мире w интерпретации $\mathcal{I} = (W, \preceq, L)$ ($\mathcal{I}, w \models_{\text{i}} \varphi$) задаётся так:

- $\mathcal{I}, w \models_{\text{i}} \mathtt{x}$, где $\mathtt{x} \in \text{Var} \quad \Leftrightarrow \quad \mathtt{x} \in L(w)$
- $\mathcal{I}, w \models_{\text{i}} \varphi \,\&\, \psi \quad \Leftrightarrow \quad \mathcal{I}, w \models_{\text{i}} \varphi$ и $\mathcal{I}, w \models_{\text{i}} \psi$
- $\mathcal{I}, w \models_{\text{i}} \varphi \vee \psi \quad \Leftrightarrow \quad \mathcal{I}, w \models_{\text{i}} \varphi$ или $\mathcal{I}, w \models_{\text{i}} \psi$
- $\mathcal{I}, w \models_{\text{i}} \neg\varphi \quad \Leftrightarrow \quad$ для любого мира w', такого что $w \preceq w'$, верно $\mathcal{I}, w' \not\models_{\text{i}} \varphi$
  - Относительно семантики К.-Б.-Г.: в модели принципиально невозможно приобрести такие умения, которые позволят решить φ
- $\mathcal{I}, w \models_{\text{i}} \varphi \to \psi \quad \Leftrightarrow \quad$ для любого мира w', такого что $w \preceq w'$ и $\mathcal{I}, w' \models_{\text{i}} \varphi$, верно и $\mathcal{I}, w' \models_{\text{i}} \psi$
  - Относительно семантики К.-Б.-Г.: согласно модели принципиально невозможно приобрести такой набор умений, чтобы научиться решать задачу φ и при этом не научиться решать ψ

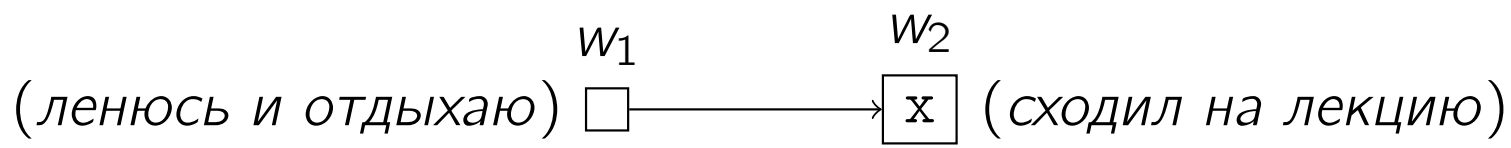
Формула φ интуиционистски общезначима ($\models_{\text{i}} \varphi$), если для любой интуиционистской интерпретации $\mathcal{I}$ и любого её мира w верно $\mathcal{I}, w \models_{\text{i}} \varphi$

Свойства интуиционистской логики

Утверждение. **Для любой переменной** $\mathtt{x}$ **верно** $\not\models_\mathtt{i} \mathtt{x} \vee \neg\mathtt{x}$

Доказательство.

Рассмотрим такую интерпретацию $\mathcal{I} = (\{w_1, w_2\}, \preceq, L)$ над $\{\mathtt{x}, \dots\}$:



- $\mathtt{x} \notin L(w_1)$
  - По семантике $\mathtt{x}$: $\mathcal{I}, w_1 \not\models_\mathtt{i} \mathtt{x}$
- $\mathtt{x} \in L(w_2)$
  - По семантике $\mathtt{x}$: $\mathcal{I}, w_2 \models_\mathtt{i} \mathtt{x}$
  - По семантике $\neg$ и соотношению $w_1 \preceq w_2$, верно $\mathcal{I}, w_1 \not\models_\mathtt{i} \neg\mathtt{x}$

Значит, по семантике $\vee$, верно $\mathcal{I}, w_1 \not\models_\mathtt{i} \mathtt{x} \vee \neg\mathtt{x}$ ▼

Свойства интуиционистской логики

Утверждение. Для любой переменной x верно $\models_\mathrm{i} \mathrm{x} \to \neg\neg\mathrm{x}$

Доказательство. Предположим *от противного*, что существуют интерпретация $\mathcal{I} = (W, \preceq, L)$ и мир w, такие что $\mathcal{I}, w \not\models_\mathrm{i} \mathrm{x} \to \neg\neg\mathrm{x}$
Тогда верно следующее:

1. Существует мир w_1, такой что $w \preceq w_1$ и, кроме того,
   1.1 $\mathcal{I}, w_1 \models_\mathrm{i} \mathrm{x}$ и
   1.2 $\mathcal{I}, w_1 \not\models_\mathrm{i} \neg\neg\mathrm{x}$ (по семантике $\to$)
2. $\mathrm{x} \in L(w_1)$ (по (1.1) и семантике x)
3. Существует мир w_2, такой что
   3.1 $w_1 \preceq w_2$ и
   3.2 $\mathcal{I}, w_2 \models \neg\mathrm{x}$ (по (1.2) и семантике $\neg$)
4. Для любого мира w_3, такого что $w_2 \preceq w_3$, верно $\mathcal{I}, w_3 \not\models_\mathrm{i} \mathrm{x}$ (по (3.2) и семантике $\neg$)
5. $\mathcal{I}, w_2 \not\models_\mathrm{i} \mathrm{x}$ (по (4) и рефлексивности отношения $\preceq$)
6. $\mathrm{x} \notin L(w_2)$ (по (5) и семантике x)
7. $\mathrm{x} \in L(w_2)$ (по (2), (3.1) и монотонности L), что *противоречит* (6) ▼

Свойства интуиционистской логики

Чтобы дать общее впечатление о свойствах интуиционистской логики, но не тратить чрезмерно много времени на детали, сформулируем напоследок несколько примечательных утверждений об интуиционистской логике без доказательства

(Кто заинтересовался, тот может попробовать доказать всё это сам)

Утверждение. **Для любых различных переменных** x **и** y **верно:**

$$\not\vdash_{\mathrm{i}} \neg\neg\mathrm{x} \to \mathrm{x}$$
$$\not\vdash_{\mathrm{i}} \neg(\mathrm{x} \,\&\, \mathrm{y}) \to (\neg\mathrm{x} \vee \neg\mathrm{y})$$
$$\not\vdash_{\mathrm{i}} \neg(\mathrm{x} \vee \mathrm{y}) \to (\neg\mathrm{x} \,\&\, \neg\mathrm{y})$$

Утверждение. **Для любых различных переменных** x, y **верно:**

$$\vdash_{\mathrm{i}} \neg\neg\neg\mathrm{x} \to \neg\mathrm{x}$$
$$\vdash_{\mathrm{i}} (\neg\mathrm{x} \vee \neg\mathrm{y}) \to \neg(\mathrm{x} \,\&\, \mathrm{y})$$
$$\vdash_{\mathrm{i}} \neg\mathrm{x} \vee \neg\neg\mathrm{x}$$

Свойства интуиционистской логики

Утверждение. **Для любых формулы φ, интуиционистской интерпретации $\mathcal{I} = (W, \preceq, L)$ и миров w_1, w_2 верно:**

$$\mathcal{I}, w_1 \models \varphi \text{ и } w_1 \preceq w_2 \quad \Rightarrow \quad \mathcal{I}, w_2 \models \varphi$$

Утверждение. **Для любой формулы φ, такой что $\models_{\text{i}} \varphi$, верно и $\models \varphi$ в логике высказываний**

Утверждение. **Для любых формул φ и ψ верно:**

$$\models_{\text{i}} \varphi \vee \psi \quad \Rightarrow \quad \models_{\text{i}} \varphi \text{ или } \models_{\text{i}} \psi$$

Математическая логика и логическое программирование

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

Блок 52

Формальная верификация

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

В математической логике есть ещё много такого,
что было бы полезно и интересно обсудить, но не получится,
так как времени до конца осталось не так уж и много

Поэтому остановимся подробно только на одном разделе:
как можно при помощи логических методов решить какую-нибудь
очень нетривиальную и очень полезную прикладную задачу,
казалось бы не связанную с логикой

Обсудим в деталях одну такую задачу из области программирования

Ошибки в программах

«**Любая нетривиальная программа содержит хотя бы одну ошибку**» (*автор неизвестен*)

Правильная программа[1] не содержит ошибок и решает задачу, которую требовалось решить при помощи этой программы

Эти утверждения выглядят разумно, но порождают ряд нетривиальных вопросов, например:

- Что такое «ошибка», насколько плохо её наличие, и нужно ли её вообще искать?
- Можно ли как-нибудь убедиться, что ошибок в программе нет и она действительно решает поставленную задачу?

1 Из утверждения выше следует, что такой программы не существует среди нетривиальных — но можно и пофантазировать

Ошибки в программах

Подход к проверке правильности работы программы,
про который знает каждый программист — это **тестирование**:

- придумывается показательный набор входных данных программы (тестовое покрытие)
- программа выполняется на тестовом покрытии
- результаты выполнения программы сравниваются с ответами к задаче, которые считаются правильными

Основной недостаток такого подхода:
ошибки всё равно остаются в программе, хотя их и становится меньше

Кроме того, в современном мире широко применяются вычислительные системы, для которых даже после «качественного» тестирования
не гарантируется отсутствие критичных ошибок:
аппаратные и программно-аппаратные, интерактивные и распределённые системы, сложные программные комплексы, ...

Ошибки в программах

В некоторых случаях протестировать код — это приемлемо (*«более-менее работает, а дальше пусть пользователь разбирается»*), но далеко не всегда: даже от небольших ошибок серьёзно зависит прибыль компаний[1], успешность проектов мирового масштаба[2], быт людей[3] и даже их жизни[4]

---

1 1994. Процессор Intel, ошибка в реализации деления чисел с плавающей точкой ⇒ замена дефектных процессоров, сотни миллионов $ убытка

2 1962–сейчас. Ракеты и спутники, взорвавшиеся и исчезнувшие из-за программных ошибок: Фобос (пропущена кавычка), Ariane 5 (ошибка в округлении чисел), Mars Global Surveyor (перепутаны английская и метрическая системы мер), Hitomi (ошибка в программе стабилизации вращения), ...

3 2003. Полное отключение электричества в нескольких областях США и Канады ⇐ ошибка в реализации взаимодействия программ оповещения об электрической нагрузке (race condition)

4 1980-е гг. Пять смертей при лечении рака аппаратом Therac-25 ⇐ ошибочное увеличение мощности радиационного облучения при очень редком сочетании времён выполнения параллельных подпрограмм (race condition)

Формальная верификация

Есть и другой подход к проверке правильности вычислительных систем:

- формулируется набор требований к выполнению системы, означающих «система функционирует правильно»
- **строго** доказывается или опровергается утверждение о том, что математическая модель, достаточно точно описывающая систему, удовлетворяют этим требованиям

Требования такого вида, записанные на математическом языке, называются формальной спецификацей системы, и соответствующий язык — языком спецификаций

Проверка соблюдения требований с помощью математических методов называется формальной верификацией системы

Если в качестве языка спецификаций выбрать какой-либо **логический язык**, то для проверки правильности программы можно будет использовать **логические методы**

Математическая логика и логическое программирование

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

Блок 53

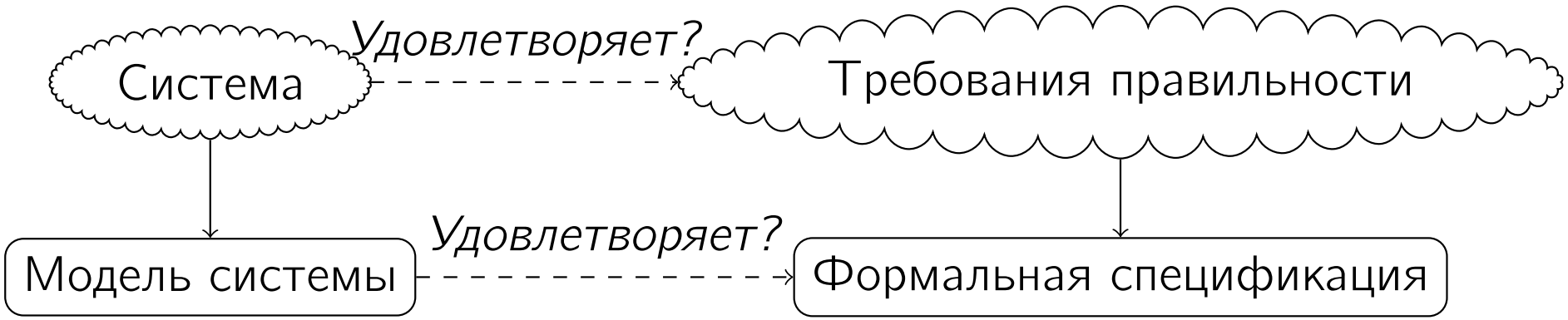
Модельные императивные программы
Постановка задачи верификации программ

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление

Формальная верификация:



Обсудим то, как можно использовать **логику предикатов** для формальной верификации **императивных программ**

- Как может быть устроена математическая модель программы?
- Как можно записывать требования к программе с использованием логики предикатов?
- Как проверить, удовлетворяет ли модель программы записанным требованиям?

Императивные программы: синтаксис

Далее считаются заданными сигнатура σ логики предикатов и множество предметных переменных Var

Синтаксис императивных программ зададим следующей БНФ:

$$
\begin{array}{lll}
\pi & ::= & \mathit{stmt} \mid \mathit{stmt}\ \pi \\
\mathit{stmt} & ::= & \emptyset \mid \qquad\qquad\qquad\qquad\qquad\qquad \text{(пустая команда)} \\
 & & \mathtt{x} := t; \mid \qquad\qquad\qquad\qquad\qquad\qquad \text{(присваивание)} \\
 & & \textbf{if } C \textbf{ then } \pi \textbf{ else } \pi \textbf{ fi} \mid \qquad \text{(ветвление)} \\
 & & \textbf{while } C \textbf{ do } \pi \textbf{ od} \qquad\qquad\qquad\qquad \text{(цикл)}
\end{array}
$$

Здесь:

- π — программа
- stmt — команда программы (или, по-другому, инструкция)
- $\mathtt{x} \in \mathrm{Var}$
- t — выражение: произвольный терм, такой что $\mathrm{Var}_t \subseteq \mathrm{Var}$
- C — условие: произвольная формула без кванторов, такая что $\mathrm{Var}_C \subseteq \mathrm{Var}$

Императивные программы: синтаксис

В примерах будут использоваться

- сигнатура, содержащая общеизвестные арифметические символы, включая
  - константы 0, 1,
  - функциональные символы $+^{(2)}$, $-^{(2)}$, $.^{(2)}$,
  - предикатные символы $=^{(2)}$, $>^{(2)}$, $\geq^{(2)}$

- арифметическая интерпретация Ar_X логики предикатов над множеством чисел X, в которой эти символы оцениваются «естественно» как соответствующие чи́сла, операции и отношения, в том числе 0, 1, $+$, $-$, $\cdot$, $=$, $>$, $\geq$ — соответственно как
  - чи́сла 0 и 1,
  - операции сложения, вычитания и умножения чисел и
  - отношения равенства чисел и их строгого и нестрогого неравенства в бо́льшую сторону

Императивные программы: синтаксис

Пример: реализация алгоритма Эвклида
вычисления наибольшего общего делителя чисел в переменных x, y

while $\neg(\mathrm{x} = \mathrm{y})$ **do**
  if $\mathrm{x} > \mathrm{y}$ **then**
    $\mathrm{x} := \mathrm{x} - \mathrm{y};$
  else
    $\mathrm{y} := \mathrm{y} - \mathrm{x};$
  fi
od

Императивные программы: операционная семантика

Значение программы — это **вычисляемая ей функция** преобразования входных данных в выходные данные

Для задания этой функции определим следующие понятия:

- Состояние данных: совокупность значений переменных, преобразуемая при выполнении программы
- Состояние управления: описание того, как текущее состояние данных будет изменяться программой в дальнейшем выполнении
- Состояние вычисления: состояние данных + состояние управления, то есть описание значений данных сейчас и в оставшейся части выполнения программы

Императивные программы: операционная семантика

Состояние данных над переменными Var в интерпретации с предметной областью D — это отображение $\sigma : \mathrm{Var} \to \mathrm{D}$
Обозначение: $[\mathtt{x}_1/\sigma(x_1), \ldots, \mathtt{x}_\mathtt{n}/\sigma(x_n)]$, если $\mathrm{Var} = \{\mathtt{x}_1, \ldots, \mathtt{x}_\mathtt{n}\}$

Состояние управления — это произвольная программа

Состояние вычисления — это пара $\langle \pi \mid \sigma \rangle$,
где π — состояние управления и σ — состояние данных

Σ — множество всех состояний данных

$\widetilde{\Sigma}$ — множество всех состояний вычисления

$\sigma\{x \leftarrow d\}$ — состояние данных, получающееся из состояния данных σ в результате присваивания переменной $\mathtt{x}$ значения d:

$$\sigma\{x \leftarrow d\}(\mathtt{x}) = d$$
$$\sigma\{x \leftarrow d\}(\mathtt{y}) = \sigma(\mathtt{y}), \text{ если } \mathtt{y} \neq \mathtt{x}$$

Императивные программы: операционная семантика

Шаг выполнения программы в интерпретации $\mathcal{I}$ описывается двуместным отношением переходов $\xrightarrow{\mathcal{I}}$ на множестве $\widetilde{\Sigma}$, состоящим из всех пар следующего вида:

- $\langle \mathrm{x} := t; \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \emptyset \mid \sigma\{x \leftarrow t\sigma\} \rangle$
- $\langle \textbf{if } C \textbf{ then } \pi_1 \textbf{ else } \pi_2 \textbf{ fi} \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \pi_1 \mid \sigma \rangle$, если $\mathcal{I} \models C\sigma$
- $\langle \textbf{if } C \textbf{ then } \pi_1 \textbf{ else } \pi_2 \textbf{ fi} \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \pi_2 \mid \sigma \rangle$, если $\mathcal{I} \not\models C\sigma$
- $\langle \textbf{while } C \textbf{ do } \pi \textbf{ od} \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \emptyset \mid \sigma \rangle$, если $\mathcal{I} \not\models C\sigma$
- $\langle \textbf{while } C \textbf{ do } \pi \textbf{ od} \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \pi \textbf{ while } C \textbf{ do } \pi \textbf{ od} \mid \sigma \rangle$, если $\mathcal{I} \models C\sigma$
- $\langle \pi_1\ \pi_2 \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \pi_1'\ \pi_2 \mid \sigma' \rangle$, если $\langle \pi_1 \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \pi_1' \mid \sigma' \rangle$
- $\langle \emptyset\ \pi \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \pi \mid \sigma \rangle$

Императивные программы: операционная семантика

Трасса программы π из состояния данных σ в интерпретации $\mathcal{I}$ — это последовательность состояний вычисления вида

$$\langle \pi \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \pi_1 \mid \sigma_1 \rangle \xrightarrow{\mathcal{I}} \langle \pi_2 \mid \sigma_2 \rangle \xrightarrow{\mathcal{I}} \dots$$

Вычислениями программы называются бесконечные трассы и трассы, оканчивающиеся состоянием управления $\emptyset$

Последнее состояние данных конечной трассы называется результатом этой трассы

Запись $\langle \pi \mid \sigma \rangle \xrightarrow{\mathcal{I}}{}^{*} \widetilde{\sigma}$ будет означать, что существует конечная трасса программы π из состояния данных σ в интерпретации $\mathcal{I}$, оканчивающаяся состоянием вычисления $\widetilde{\sigma}$

Программой π в интерпретации $\mathcal{I}$ вычисляется частичная функция $\mathcal{I}[\pi] : \Sigma \to \Sigma$ следующего вида:

$$\mathcal{I}[\pi](\sigma) = \sigma' \quad \Leftrightarrow \quad \langle \pi \mid \sigma \rangle \xrightarrow{\mathcal{I}}{}^{*} \langle \emptyset \mid \sigma' \rangle$$

Императивные программы: операционная семантика (пример)

$$\mathrm{Var} = \{\mathrm{x}, \mathrm{y}\}$$

$$\pi = \textbf{while } \neg(\mathrm{x} = \mathrm{y}) \textbf{ do if } \mathrm{x} > \mathrm{y} \textbf{ then } \mathrm{x} := \mathrm{x} - \mathrm{y}; \textbf{ else } \mathrm{y} := \mathrm{y} - \mathrm{x}; \textbf{ fi od}$$

Вычисление π из $[\mathrm{x}/2, \mathrm{y}/4]$ в $Ar_{\mathbb{Z}}$, где $\mathbb{Z}$ — множество всех целых чисел:

$$\langle \pi \mid [\mathrm{x}/2, \mathrm{y}/4] \rangle$$

$$Ar_{\mathbb{Z}} \downarrow \quad \text{т.к. } Ar_{\mathbb{Z}} \models \neg(\mathrm{x} = \mathrm{y})[\mathrm{x}/2, \mathrm{y}/4]$$

$$\langle \textbf{if } \mathrm{x} > \mathrm{y} \textbf{ then } \mathrm{x} := \mathrm{x} - \mathrm{y}; \textbf{ else } \mathrm{y} := \mathrm{y} - \mathrm{x}; \textbf{ fi } \pi \mid [\mathrm{x}/2, \mathrm{y}/4] \rangle$$

$$Ar_{\mathbb{Z}} \downarrow \quad \text{т.к. } Ar_{\mathbb{Z}} \not\models (\mathrm{x} > \mathrm{y})[\mathrm{x}/2, \mathrm{y}/4]$$

$$\langle \mathrm{y} := \mathrm{y} - \mathrm{x}; \pi \mid [\mathrm{x}/2, \mathrm{y}/4] \rangle$$

$$Ar_{\mathbb{Z}} \downarrow \quad \text{т.к. } [\mathrm{x}/2, \mathrm{y}/4]\{\mathrm{y} \leftarrow (\mathrm{y} - \mathrm{x})[\mathrm{x}/2, \mathrm{y}/4]\} = [\mathrm{x}/2, \mathrm{y}/2]$$

$$\langle \emptyset\ \pi \mid [\mathrm{x}/2, \mathrm{y}/2] \rangle$$

$$Ar_{\mathbb{Z}} \downarrow$$

$$\langle \pi \mid [\mathrm{x}/2, \mathrm{y}/2] \rangle$$

$$Ar_{\mathbb{Z}} \downarrow \quad \text{т.к. } Ar_{\mathbb{Z}} \not\models \neg(\mathrm{x} = \mathrm{y})[\mathrm{x}/2, \mathrm{y}/2]$$

$$\langle \emptyset \mid [\mathrm{x}/2, \mathrm{y}/2] \rangle$$

Результат этого вычисления: $[\mathrm{x}/2, \mathrm{y}/2]$

Следовательно, $Ar_{\mathbb{Z}}[\pi]([\mathrm{x}/2, \mathrm{y}/4]) = [\mathrm{x}/2, \mathrm{y}/2]$

Задача верификации программ

Требования правильности выполнения программы могут быть записаны как два отношения на состояниях данных:

- предусловие φ,
  задающее общий вид допустимых входных данных
- постусловие ψ,
  описывающее устройство правильных выходных данных

Принято рассматривать два вида правильности выполнения программы относительно заданных предусловия и постусловия:

- частичная корректность: результат любого конечного вычисления программы на допустимых входных данных правилен
- полная корректность: любое вычисление программы на допустимых входных данных конечно, и результат этого вычисления правилен

Остановимся подробнее на частичной корректности программ

Задача верификации программ

Тройка Хоара (по-другому — триплет Хоара) — это запись вида $\{\varphi\}\pi\{\psi\}$, где

- φ — формула логики предикатов, называемая предусловием
- π — программа
- ψ — формула логики предикатов, называемая постусловием

Триплет $\{\varphi\}\pi\{\psi\}$ истинен в интерпретации $\mathcal{I}$ ($\mathcal{I} \models \{\varphi\}\pi\{\psi\}$), если для любых состояний данных σ, σ' верно следующее:

если $\mathcal{I} \models \varphi\sigma$ и значение $\sigma' = \mathcal{I}[\pi](\sigma)$ определено, то $\mathcal{I} \models \psi\sigma'$

Программа π частично корректна в интерпретации $\mathcal{I}$ относительно предусловия φ и постусловия ψ, если $\mathcal{I} \models \{\varphi\}\pi\{\psi\}$

Задача верификации императивных программ:
для заданных программы π, предусловия φ, постусловия ψ и интерпретации $\mathcal{I}$ проверить справедливость соотношения $\mathcal{I} \models \{\varphi\}\pi\{\psi\}$

Математическая логика и логическое программирование

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

Блок 54

Логика Хоара
Автоматизация проверки правильности программ

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

ВМК МГУ, 2023/2024, осенний семестр

Логика Хоара

Для проверки частичной корректности
модельных императивных программ
можно адаптировать метод семантических таблиц:
ввести понятие вывода (дерева, построенного по правилам вывода)
и свести проверку корректности к построению успешного вывода

Правила вывода будут выглядеть так:[1]

$$\frac{\Phi}{\Psi}, \quad \frac{\Phi}{\Psi, \Omega}, \quad \frac{\Phi}{\varphi} \quad \text{или} \quad \frac{\Phi}{\varphi, \Omega, \psi},$$

где Φ, Ψ, Ω — триплеты Хоара и φ, ψ — формулы логики предикатов

Содержательное прочтение:
если в $\mathcal{I}$ истинны все триплеты и формулы под чертой
то в $\mathcal{I}$ истинен триплет Φ

1 Hoare C.A.R. An axiomatic basis for computer programming. 1969

Логика Хоара

Вот эти правила:

$$R_{\emptyset}: \frac{\{\varphi\}\emptyset\{\varphi\}}{\mathbb{t}} \qquad R_{:=}: \frac{\{\varphi\{\mathrm{x}/t\}\}\mathrm{x} := t;\{\varphi\}}{\mathbb{t}}$$

(подстановка $\{\mathrm{x}/t\}$ правильна для φ)

$$R_{\textbf{if}}: \frac{\{\varphi\}\textbf{if } C \textbf{ then } \pi_1 \textbf{ else } \pi_2 \textbf{ fi}\{\psi\}}{\{\varphi \,\&\, C\}\pi_1\{\psi\}, \{\varphi \,\&\, \neg C\}\pi_2\{\psi\}}$$

$$R_{\textbf{while}}: \frac{\{\varphi\}\textbf{while } C \textbf{ do } \pi \textbf{ od}\{\varphi \,\&\, \neg C\}}{\{\varphi \,\&\, C\}\pi\{\varphi\}}$$

$$R_{seq}: \frac{\{\varphi\}\pi_1\ \pi_2\{\psi\}}{\{\varphi\}\pi_1\{\chi\}, \{\chi\}\pi_2\{\psi\}}$$

$$R_{inf}: \frac{\{\varphi\}\pi\{\psi\}}{\varphi \to \varphi', \{\varphi'\}\pi\{\psi'\}, \psi' \to \psi}$$

Логика Хоара

Лемма (о корректности правил вывода Хоара)

Для любой интерпретации $\mathcal{I}$ и любого из правил

$$R_{\emptyset},\ R_{:=},\ R_{\textbf{if}},\ R_{\textbf{while}},\ R_{seq},\ R_{while}$$

$$\left(\frac{\Phi}{\Psi},\quad \frac{\Phi}{\Psi,\Omega},\quad \frac{\Phi}{\varphi},\quad \frac{\Phi}{\varphi,\Psi,\psi}\right)$$

верно следующее: если $\mathcal{I} \models \Psi$, $\mathcal{I} \models \Omega$, $\mathcal{I} \models \varphi$ и $\mathcal{I} \models \psi$, то $\mathcal{I} \models \Phi$

Доказательство.

Подробно рассмотрим только правило $R_{:=}$: $\dfrac{\{\varphi\{\mathrm{x}/t\}\}\mathrm{x} := t;\{\varphi\}}{\mathbb{t}}$

Пусть σ — произвольное состояние данных, такое что $\mathcal{I} \models \varphi\{\mathrm{x}/t\}\sigma$

Шаг вычисления для программы «$\mathrm{x} := t;$» устроен так:

$$\langle \mathrm{x} := t; \mid \sigma\rangle \xrightarrow{\mathcal{I}} \langle \emptyset \mid \sigma\{x \leftarrow t\sigma\}\rangle$$

Так как $\mathcal{I} \models \varphi\{\mathrm{x}/t\}\sigma$, то верно и $\mathcal{I} \models \varphi(\sigma\{x \leftarrow t\sigma\})$

Следовательно, $\mathcal{I} \models \{\varphi\{\mathrm{x}/t\}\}\mathrm{x} := t;\{\varphi\}$

Корректность остальных правил обосновывается по той же схеме ▼

Логика Хоара

Вывод триплета $\{\varphi\}\pi\{\psi\}$ — это дерево следующего вида:

- вершины размечены триплетами и формулами
- корень помечен триплетом $\{\varphi\}\pi\{\psi\}$
- дети вершины определяются относительно правил логики Хоара так же, как и в дереве табличного вывода относительно правил табличного вывода
- все листья помечены формулами

Успешный вывод триплета $\{\varphi\}\pi\{\psi\}$ в интерпретации $\mathcal{I}$ — это конечный вывод, все листья которого помечены формулами, истинными в $\mathcal{I}$

Логика Хоара

Теорема (о корректности логики Хоара)
Если существует успешный вывод триплета $\{\varphi\}\pi\{\psi\}$ в интерпретации $\mathcal{I}$, то $\mathcal{I} \models \{\varphi\}\pi\{\psi\}$

Доказательство.

Рассмотрим произвольный успешный вывод D для $\{\varphi\}\pi\{\psi\}$ в $\mathcal{I}$

Во всех листьях D записаны формулы, истинные в $\mathcal{I}$

Применив лемму о корректности правил вывода конечное число раз, получим истинность триплета $\{\varphi\}\pi\{\psi\}$ в $\mathcal{I}$ ▼

Логика Хоара

Пример: алгоритм Эвклида

$\pi =$ **while** $\neg(\mathtt{x} = \mathtt{y})$ **do if** $\mathtt{x} > \mathtt{y}$ **then** $\mathtt{x} := \mathtt{x} - \mathtt{y}$; **else** $\mathtt{y} := \mathtt{y} - \mathtt{x}$; **fi od**

Докажем, что в результате выполнения π в интерпретации $Ar_{\mathbb{Z}}$ в $\mathtt{x}$ записывается **наибольший общий делитель** (НОД) положительных чисел, заданных в $\mathtt{x}$ и $\mathtt{y}$ перед выполнением

Предусловие φ: чи́сла $\mathtt{x}$ и $\mathtt{y}$ положительны, и $\mathtt{z}$ — переменная, обозначающая НОД $\mathtt{x}$ и $\mathtt{y}$ в начале выполнения программы

$$\begin{aligned} \mathtt{u}|\mathtt{v} = & \quad \exists \mathtt{w}\,(\mathtt{v} = \mathtt{u} \cdot \mathtt{w}) \\ \gcd(\mathtt{u},\mathtt{v},\mathtt{w}) = & \quad (\mathtt{w}|\mathtt{u}) \,\&\, (\mathtt{w}|\mathtt{v}) \,\&\, \forall \mathtt{r}\,((\mathtt{r}|\mathtt{u}) \,\&\, (\mathtt{r}|\mathtt{v}) \to (\mathtt{w} \geq \mathtt{r})) \\ \varphi = & \quad \mathtt{x} > 0 \,\&\, \mathtt{y} > 0 \,\&\, \gcd(\mathtt{x},\mathtt{y},\mathtt{z}) \end{aligned}$$

Постусловие ψ: в $\mathtt{x}$ записан требуемый НОД

$$\psi = (\mathtt{x} = \mathtt{z})$$

Для обоснования правильности π достаточно построить успешный табличный вывод для триплета $\{\varphi\}\pi\{\psi\}$

Логика Хоара

$\{x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z)\}$
while $\neg(x = y)$ **do if** $x > y$ **then** $x := x - y$; **else** $y := y - x$; **fi od**
$\{x = z\}$

χ_1: $x > 0 \,\&\, x > 0 \,\&\, \gcd(x, y, z) \to x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z)$

χ_2: $x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z) \,\&\, \neg\neg(x = y) \to x = z$

$\{x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z)\}$
while $\neg(x = y)$ **do if** $x > y$ **then** $x := x - y$; **else** $y := y - x$; **fi od**
$\{x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z) \,\&\, \neg\neg(x = y)\}$

$$R_{inf}: \frac{\{\varphi\}\pi\{\psi\}}{\varphi \to \varphi', \{\varphi'\}\pi\{\psi'\}, \psi' \to \psi}$$

$$\mathcal{I} \models \chi_1$$
$$\mathcal{I} \models \chi_2$$

Логика Хоара

$\{x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z)\}$
while $\neg(x = y)$ **do if** $x > y$ **then** $x := x - y$; **else** $y := y - x$; **fi od**
$\{x = z\}$

χ_1: $x > 0 \,\&\, x > 0 \,\&\, \gcd(x, y, z) \to x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z)$

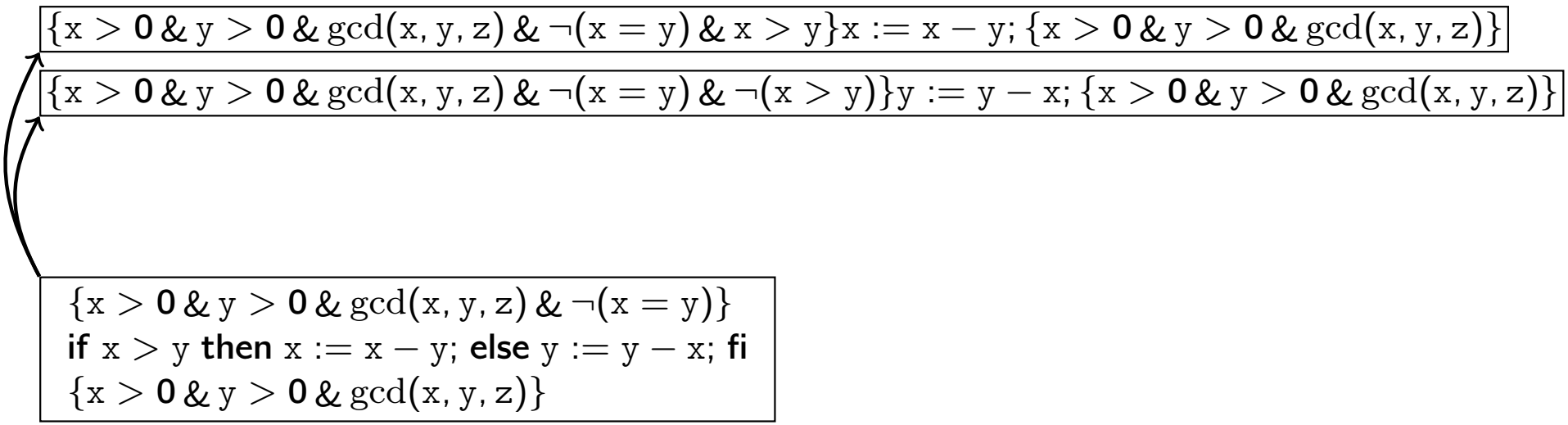
χ_2: $x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z) \,\&\, \neg\neg(x = y) \to x = z$

$\{x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z)\}$
while $\neg(x = y)$ **do if** $x > y$ **then** $x := x - y$; **else** $y := y - x$; **fi od**
$\{x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z) \,\&\, \neg\neg(x = y)\}$

$\{x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z) \,\&\, \neg(x = y)\}$
if $x > y$ **then** $x := x - y$; **else** $y := y - x$; **fi**
$\{x > 0 \,\&\, y > 0 \,\&\, \gcd(x, y, z)\}$

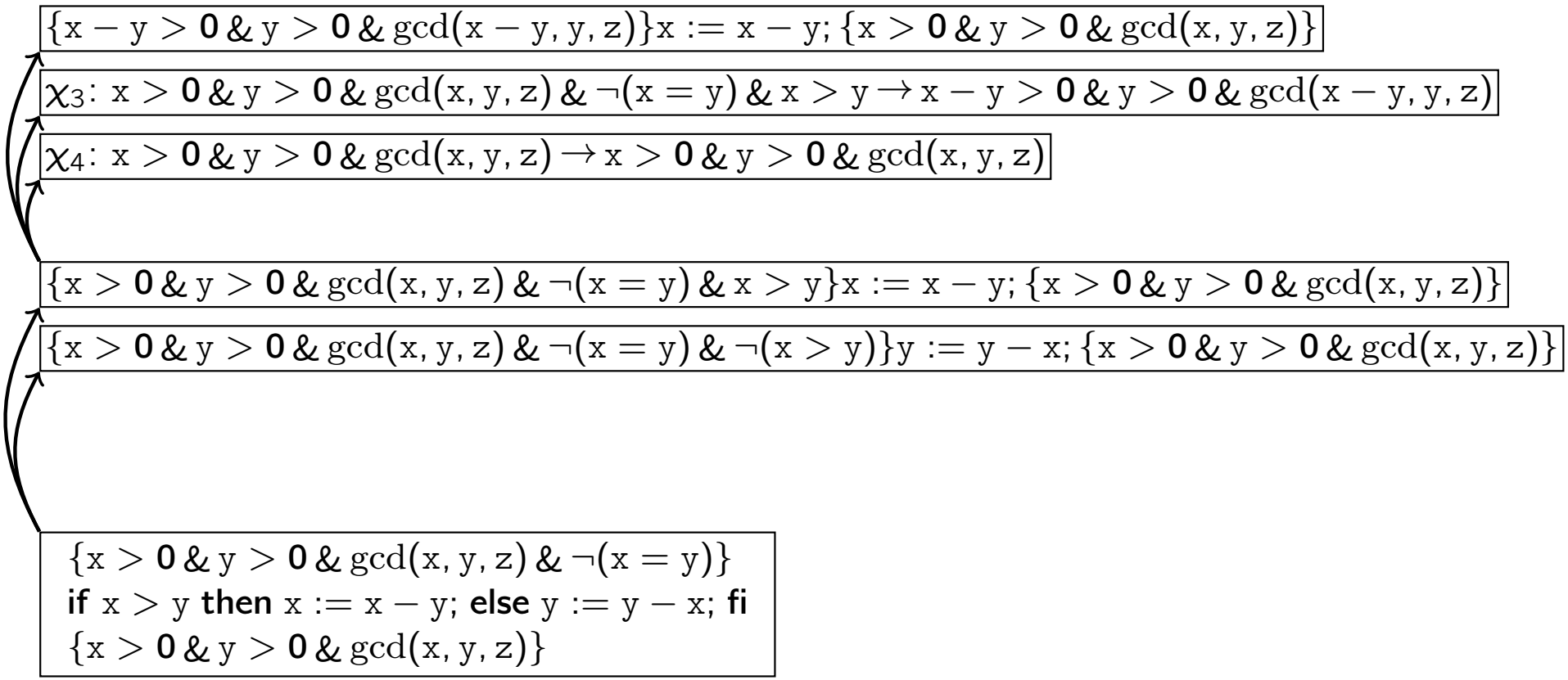
$$R_{\textbf{while}}\colon \frac{\{\varphi\}\textbf{while } C \textbf{ do } \pi \textbf{ od}\{\varphi \,\&\, \neg C\}}{\{\varphi \,\&\, C\}\pi\{\varphi\}}$$

Логика Хоара



$$R_{\textbf{if}}\colon \frac{\{\varphi\}\textbf{if } C \textbf{ then } \pi_1 \textbf{ else } \pi_2 \textbf{ fi}\{\psi\}}{\{\varphi \,\&\, C\}\pi_1\{\psi\}, \{\varphi \,\&\, \neg C\}\pi_2\{\psi\}}$$

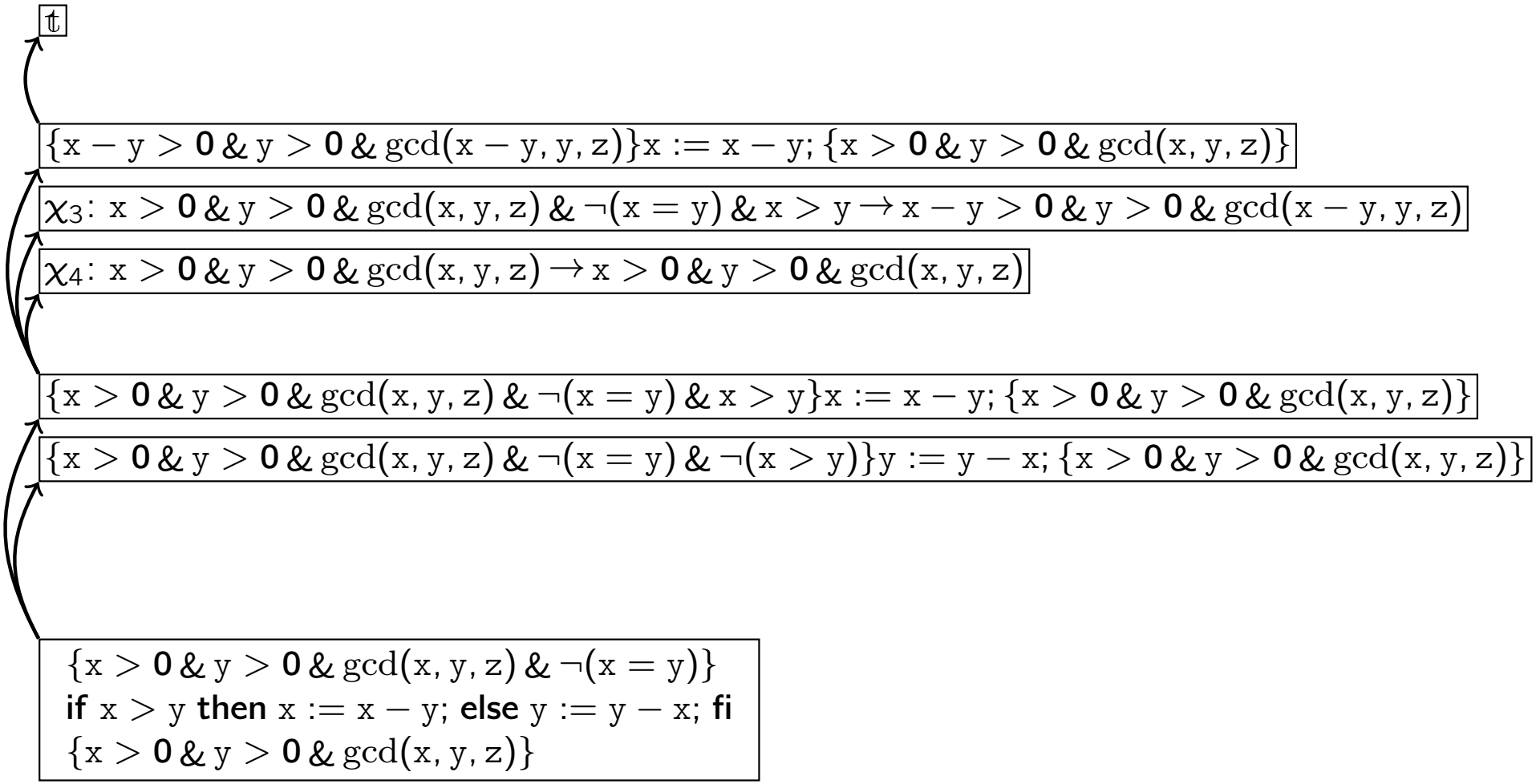
Логика Хоара



$$R_{inf}\colon \frac{\{\varphi\}\pi\{\psi\}}{\varphi \to \varphi', \{\varphi'\}\pi\{\psi'\}, \psi' \to \psi}$$

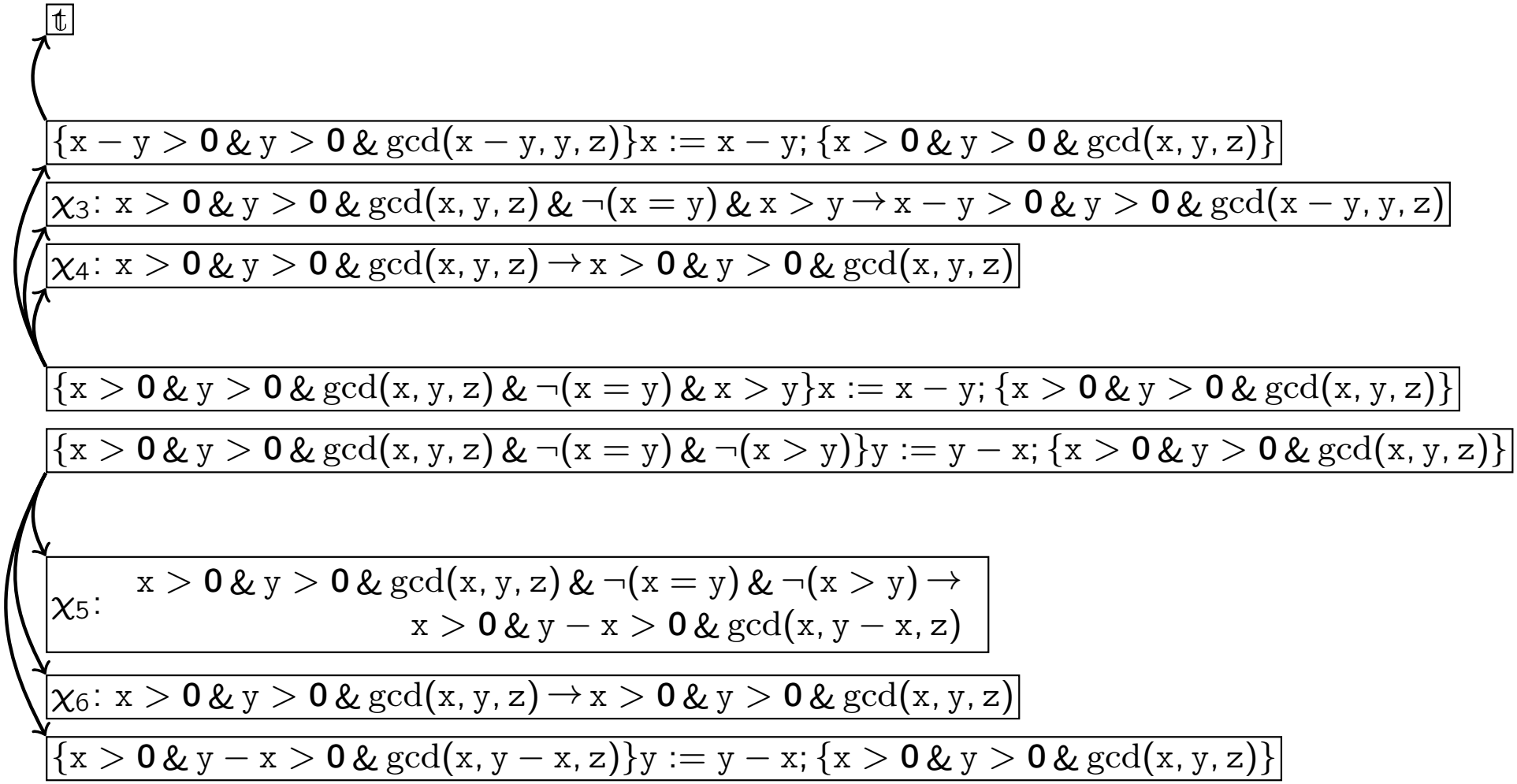
$$\mathcal{I} \models \chi_3$$
$$\mathcal{I} \models \chi_4$$

Логика Хоара



$$R_{:=}\colon \frac{\{\varphi\{x/t\}\}x := t; \{\varphi\}}{\mathbb{t}}$$

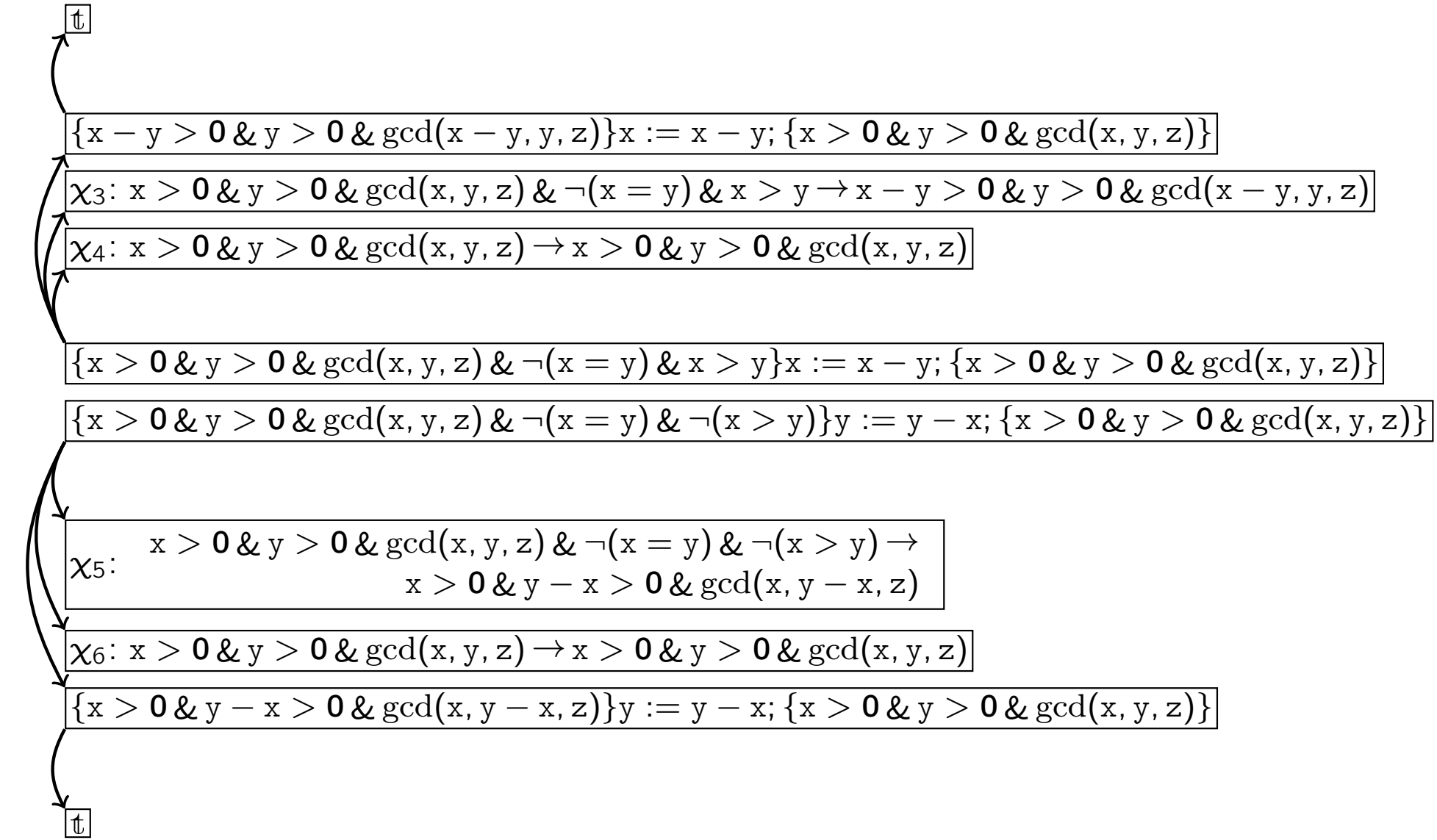
Логика Хоара



$$R_{inf}\colon \frac{\{\varphi\}\pi\{\psi\}}{\varphi \to \varphi', \{\varphi'\}\pi\{\psi'\}, \psi' \to \psi}$$

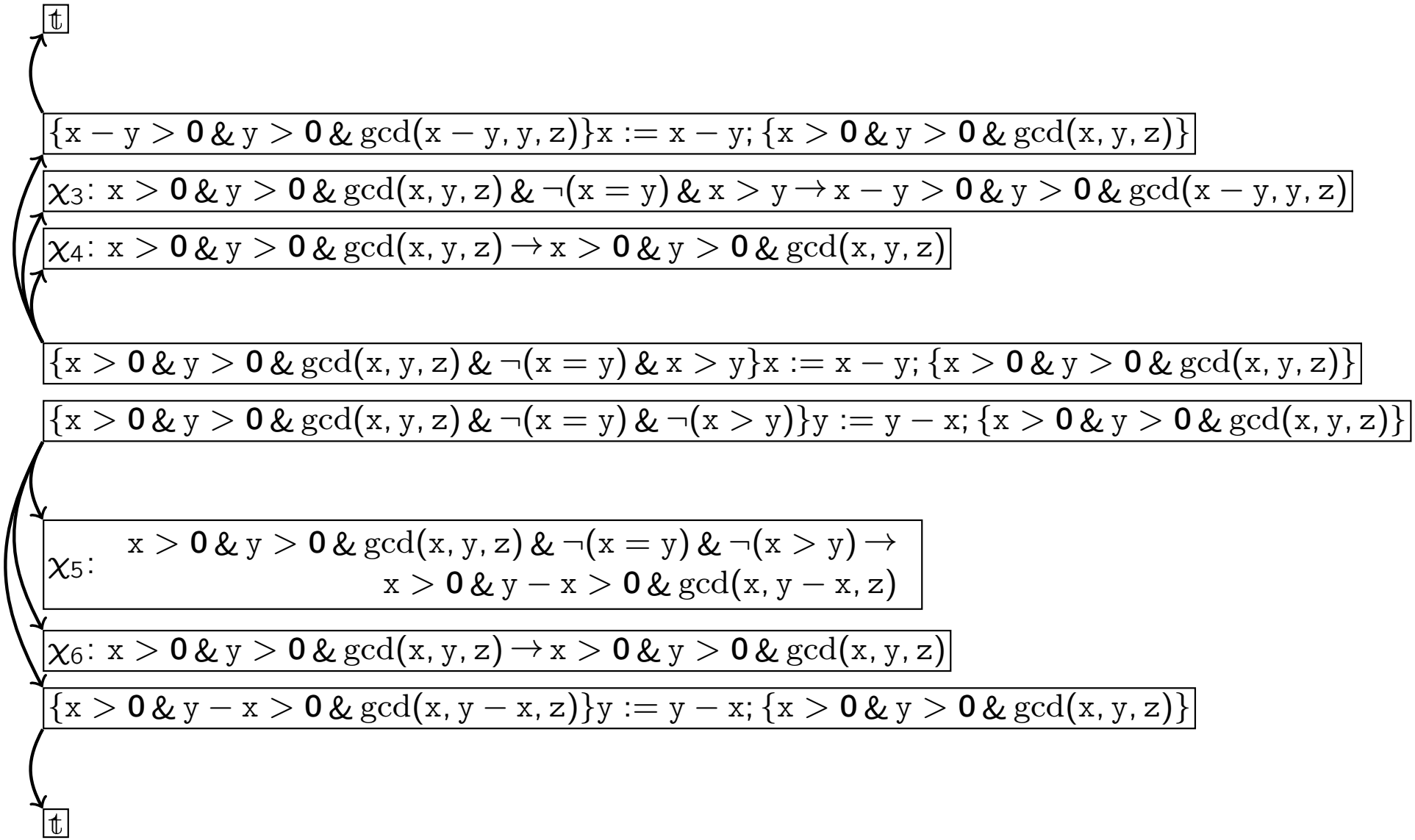
$$\mathcal{I} \models \chi_5$$
$$\mathcal{I} \models \chi_6$$

Логика Хоара



$$R_{:=}\colon \frac{\{\varphi\{x/t\}\}x := t;\{\varphi\}}{\mathbb{t}}$$

Логика Хоара



Частичная корректность программы доказана

Полнота логики Хоара

Согласно теореме о корректности логики Хоара,
правильность программы можно обосновать, построив
успешный табличный вывод подходящего триплета Хоара

А правда ли, что корректность **любой** правильной программы
может быть обоснована построением
успешного табличного вывода подходящего триплета?

На самом деле это не один вопрос, а два принципиально разных:

1. Все ли свойства правильности программ
   могут быть записаны в виде формул логики предикатов?
2. Для любого ли истинного триплета существует успешный вывод?

Полнота логики Хоара

Ответ на оба вопроса существенно зависит от сигнатуры,
в которой записываются предусловия и постусловия

Если эта сигнатура слишком «скудная», то:

- Её может быть недостаточно
  для записи свойств правильности программы
- При применении правил
  $$R_{seq} : \frac{\{\varphi\}\pi_1\ \pi_2\{\psi\}}{\{\varphi\}\pi_1\{\chi\}, \{\chi\}\pi_2\{\psi\}} \qquad R_{inf} : \frac{\{\varphi\}\pi\{\psi\}}{\varphi \to \varphi', \{\varphi'\}\pi\{\psi'\}, \psi' \to \psi}$$
  может и не найтись подходящих формул φ', ψ', χ,
  позволяющих достроить вывод до успешного

При этом чем «богаче» сигнатура, тем труднее анализировать
истинность формул и подбирать «подходящие» формулы при
построении вывода

Автоматизация проверки правильности программ

А если сигнатура достаточно «богата», то можно ли реализовать программу, автоматически доказывающую корректность программ?

Как и в вопросе про полноту, это на самом деле не один вопрос, а два:

1. Можно ли автоматизировать запись триплетов Хоара, отвечающих свойствам правильности программ?
2. Можно ли автоматизировать построение успешного вывода для заданного триплета Хоара?

Ответ на первый вопрос однозначен — **нет**:
один из главных недостатков формальной верификации состоит в том, что формальная спецификация программы, как правило, создаётся вручную специально обученным экспертом

Автоматизация проверки правильности программ

С автоматизацией построения успешного вывода
для заданного триплета Хоара дела обстоят чуть лучше

Слабейшим предусловием для программы π и постусловия ψ
в интерпретации $\mathcal{I}$ называется формула $wpr(\pi, \psi, \mathcal{I})$, такая что

- $\mathcal{I} \models \{wpr(\pi, \psi, \mathcal{I})\}\pi\{\psi\}$ и
- для любой формулы φ, такой что $\mathcal{I} \models \{\varphi\}\pi\{\psi\}$,
  верно соотношение $\mathcal{I} \models \varphi \to wpr(\pi, \psi, \mathcal{I})$

Слабейших предусловий на самом деле может быть много,
но они имеют одинаковый смысл в $\mathcal{I}$,
так что для простоты иногда будем считать, что оно единственно

Теорема. $\mathcal{I} \models \{\varphi\}\pi\{\psi\} \Leftrightarrow$
$\mathcal{I} \models \{wpr(\pi, \psi, \mathcal{I})\}\pi\{\psi\}$ **и** $\mathcal{I} \models \varphi \to wpr(\pi, \psi, \mathcal{I})$

Доказательство. Следует из определения слабейшего предусловия ▼

Автоматизация проверки правильности программ

Теорема (о слабейшем предусловии)

- $wpr(\emptyset, \psi, \mathcal{I}) = \psi$
- $wpr(\mathrm{x} := t;, \psi, \mathcal{I}) = \psi\{\mathrm{x}/t\}$,
  если подстановка $\{\mathrm{x}/t\}$ **правильна для** ψ
- $wpr(\textbf{if } C \textbf{ then } \pi_1 \textbf{ else } \pi_2 \textbf{ fi}, \psi, \mathcal{I}) =$
  $C \,\&\, wpr(\pi_1, \psi, \mathcal{I}) \vee \neg C \,\&\, wpr(\pi_2, \psi, \mathcal{I})$
- $wpr(\pi_1\ \pi_2, \psi, \mathcal{I}) = wpr(\pi_1, wpr(\pi_2, \psi, \mathcal{I}), \mathcal{I})$

Доказательство опустим, чтобы сэкономить время

Таким образом,

- проверка корректности программы сводится к вычислению слабейшего предусловия и проверки истинности заданной формулы
- для программ **без циклов** слабейшее предусловие не зависит от выбора интерпретации и вычисляется очень просто

Автоматизация проверки правильности программ

А как вычислить слабейшее предусловие для цикла?

Устройство слабейшего предусловия тесно связано с устройством правил доказательства корректности программ:
вычисление этого предусловия — настолько же (не)простая задача, насколько и применение правила для построения успешного вывода

Чтобы применить правило

$$R_{\textbf{while}} : \frac{\{\varphi\}\textbf{while } C \textbf{ do } \pi \textbf{ od}\{\varphi \,\&\, \neg C\}}{\{\varphi \,\&\, C\}\pi\{\varphi\}},$$

требуется предварительно найти формулу φ, реализующую свойство состояний данных, сохраняющееся при выполнении каждого витка цикла

Эта формула φ называется инвариантом цикла и явно или неявно используется практически всегда при анализе циклов

Автоматическая генерация инвариантов циклов — это **ключевая проблема** автоматизации проверки правильности программ

Математическая логика и логическое программирование

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

Блок 55

Проверка правильности распределённых систем
Пара слов о методе проверки моделей

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

ВМК МГУ, 2023/2024, осенний семестр

Проверка правильности распределённых систем

Начнём с примера: рассмотрим две процедуры в синтаксисе языка C:

```
void стипендия() {
  счёт += 1 000;
}
```

```
void надбавка() {
  счёт += 1 000 000;
}
```

Если они выполняются последовательно в каком-либо порядке, то можно легко убедиться (например, при помощи логики Хоара), что стипендия с надбавкой начисляются корректно:

$$\{\text{счёт} = \text{x}\}$$
$$\text{счёт} := \text{счёт} + 1\,000;$$
$$\text{счёт} := \text{счёт} + 1\,000\,000;$$
$$\{\text{счёт} = \text{x} + 1\,001\,000\}$$

$$\{\text{счёт} = \text{x}\}$$
$$\text{счёт} := \text{счёт} + 1\,000\,000;$$
$$\text{счёт} := \text{счёт} + 1\,000;$$
$$\{\text{счёт} = \text{x} + 1\,001\,000\}$$

Проверка правильности распределённых систем

Начнём с примера: рассмотрим две процедуры в синтаксисе языка C:

```
void стипендия() {
  счёт += 1 000;
}
```

```
void надбавка() {
  счёт += 1 000 000;
}
```

На практике каждое из присваиваний может выполняться
и за несколько шагов (действий) — например:

1. Чтение значения аргумента-переменной
2. Прибавление константы к прочитанному значению
3. Запись результата в переменную

Для последовательно выполняющихся процедур
степень детализации семантики неважна, и ответ о корректности,
полученный при помощи логики Хоара, можно считать правильным

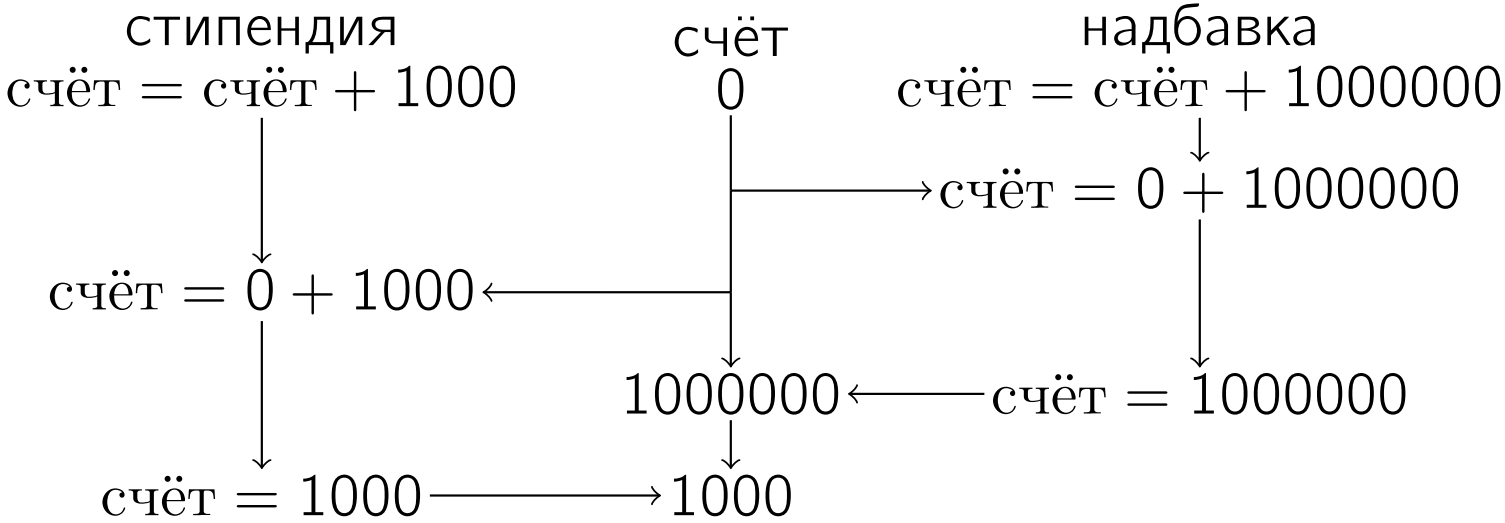
Проверка правильности распределённых систем

Начнём с примера: рассмотрим две процедуры в синтаксисе языка C:

```
void стипендия() {
  счёт += 1 000;
}
```

```
void надбавка() {
  счёт += 1 000 000;
}
```

Но если эти процедуры будут выполняться **параллельно**,
то ответ, полученный при помощи логики Хоара,
окажется не соответствующим реальности:



Честно заработанная надбавка исчезла со счёта
из-за неудачного редкого стечения технических обстоятельств

Проверка правильности распределённых систем

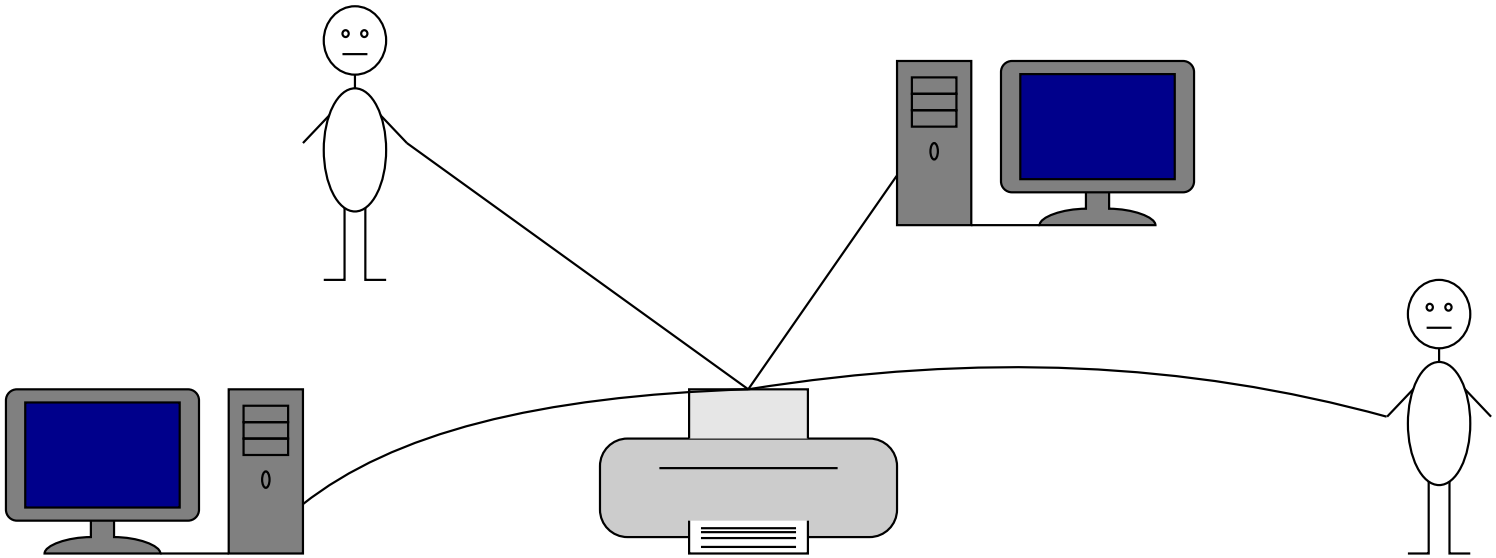
Такую ошибку нетрудно «проглядеть» при разработке программы

При этом её можно считать невоспроизводимой
и практически необнаружимой при помощи тестирования:

- Чтобы ошибка возникла, присваивания должны выполниться *почти одновременно* — настолько, чтобы третий шаг одного из них выполнился строго между первым и третьим шагами другого
- Разработчик программы обычно не может контролировать время выполнения шагов присваиваний настолько точно, чтобы можно было перемешивать шаги заранее заданным способом
- Даже если есть подходящие средства контроля точности, параллельно выполняющихся действий обычно настолько много, что нельзя считать разумным тестовое покрытие, содержащее всевозможные порядки выполнения
  - Например, для 70-ти параллельных независимых действий существует 70! порядков их выполнения, а это больше чем гугол

Проверка правильности распределённых систем

Ещё один пример



Представим себе сетевой принтер, с которым могут взаимодействовать пользователи при помощи удалённых компьютеров и находясь непосредственно у принтера

Как протестировать такую систему?

Можно ли, как-нибудь «разумно» протестировав систему, заключить, что в ней нет критичных ошибок?

Как могут быть устроены ошибки в такой системе?

Проверка правильности распределённых систем

Для проверки правильности распределённых систем, то есть таких, в которых компоненты выполняются параллельно, взаимодействуя между собой для достижения общей цели, одного только тестирования оказывается недостаточно:

- Достаточно полное тестирование зачастую чересчур трудоёмко или даже невозможно
- Протестировав каждую часть системы, как правило, нельзя быть уверенным в том, что система в целом будет работать верно
- Некоторые критичные ошибки в работе системы возникают при настолько специфичных обстоятельствах, что обнаружить их тестированием практически невозможно

Поэтому для полноценной проверки правильности распределённых систем следует уметь применять и другие подходы — например, формальную верификацию

Пара слов о методе проверки моделей

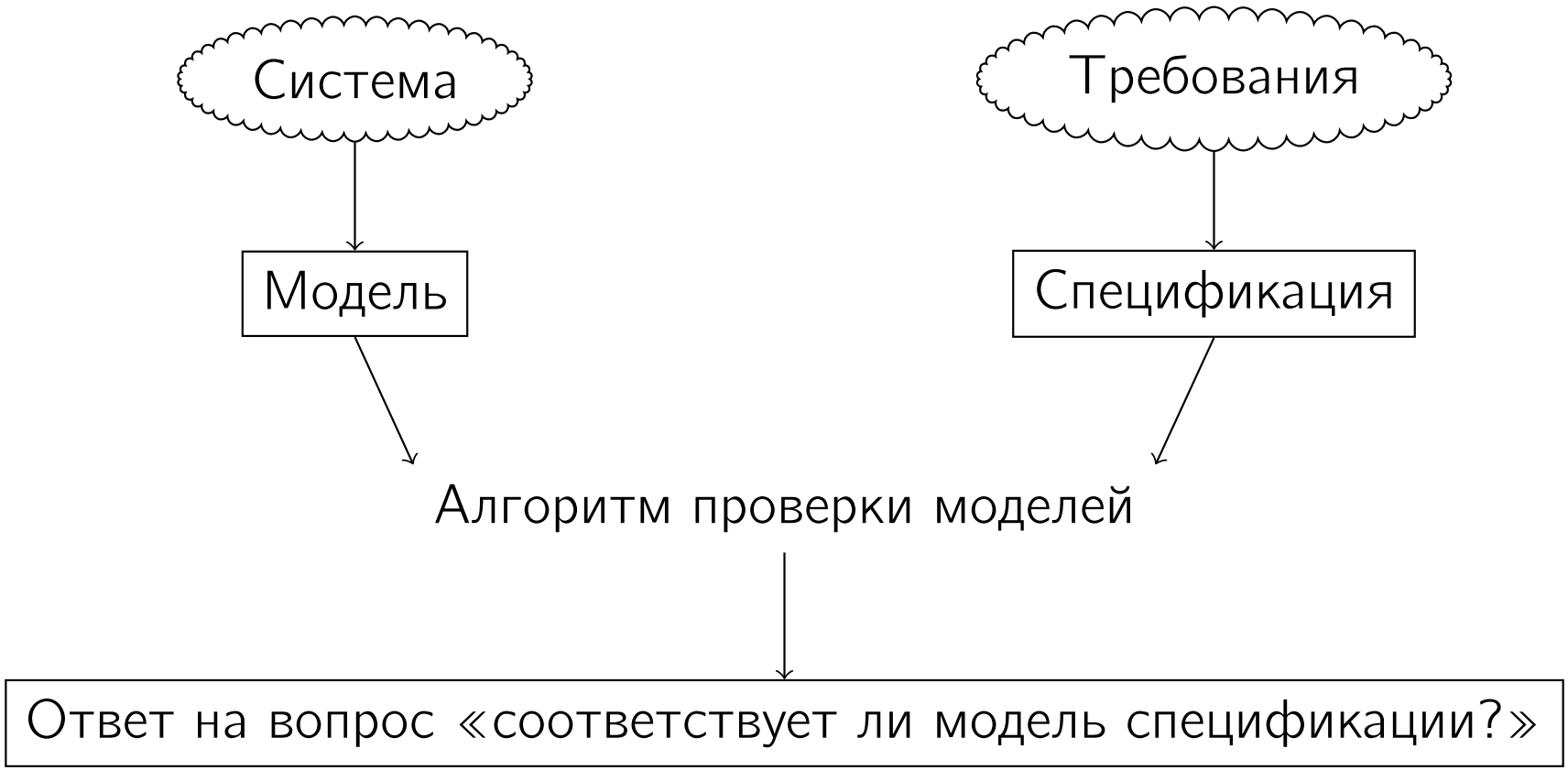
Для формальной верификации распределённых систем успешно применяется метод проверки моделей (верификации моделей программ; model checking)

Согласно этому методу:

1. Описывается модель системы,
   - достаточно детальная, чтобы интересующие свойства модели можно было перенести на исследуемую систему, и при этом
   - достаточно простая для эффективного построения и анализа
2. Выбирается язык спецификаций,
   - соответствующий устройству модели и при этом
   - достаточно выразительный для записи желаемых требований
3. Разработчик создаёт и пользователи применяют средства **автоматической** проверки того, что модель соответствует её спецификации

Пара слов о методе проверки моделей

Краткая схема применения метода проверки моделей



Напоследок обсу́дим одну из «базовых» конкретных постановок задачи проверки моделей и решение этой задачи

Математическая логика и логическое программирование

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

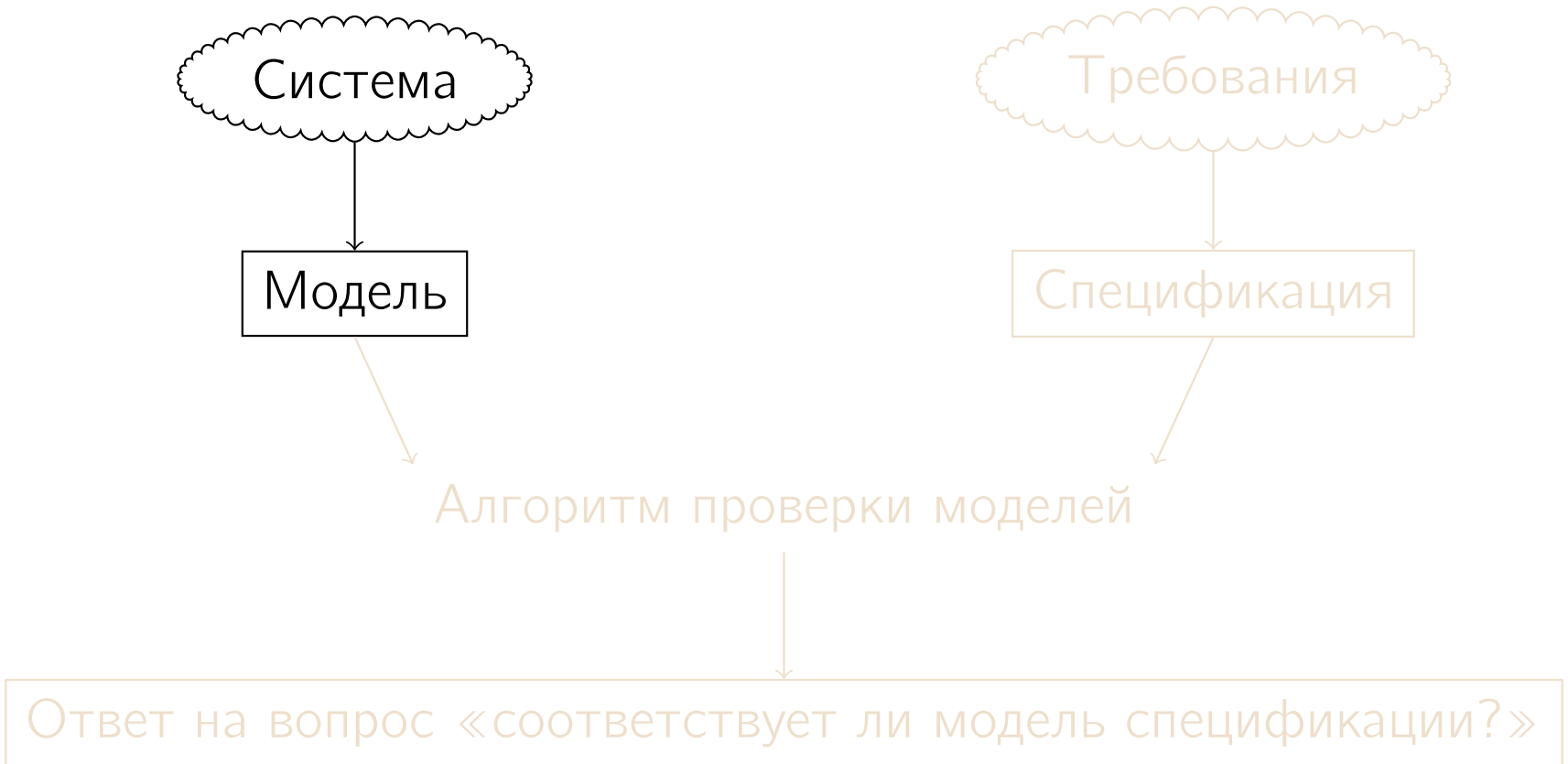
Блок 56

Размеченные системы переходов

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление (краткая схема проверки моделей)



Размеченные системы переходов

Чтобы лучше понять, как (*и почему именно так*) устроена модель вычислительной системы, используемая в проверке моделей, рассмотрим **для примера** такую систему

Система состоит из кофейного автомата и покупателя

Кофейный автомат имеет приёмник монет и кнопки «чай» и «кофе» и запрограммирован на такое поведение:

- В режиме ожидания приёмник монет открыт
- После приёма монеты
  - приёмник закрывается, ожидается нажатие на одну из кнопок,
  - после нажатия на кнопку соответствующий напиток выдаётся покупателю, монета удаляется из приёмника и автомат переходит в режим ожидания

Покупатель в зависимости от своего желания может кидать монеты в приёмник и нажимать на кнопки

Размеченные системы переходов

Поведение кофейного автомата можно представить себе так

В каждый момент времени он находится в некотором состоянии:
«ожидает монету», «ожидает нажатия кнопки»,
«выдаёт чай», «выдаёт кофе»

Иногда автомат переходит из одного состояния в другое
согласно внешним обстоятельствам и своей программе

Некоторое состояние (начальное) отвечает запуску программы

Чтобы проверить правильность работы автомата, достаточно
выделить набор *простых* свойств состояний (атомарных высказываний)
и проанализировать изменение этих свойств с течением времени:
«приёмник открыт», «в приёмнике есть монета»,
«выдаётся чай», «выдаётся кофе»

Размеченные системы переходов

AP — так будем обозначать множество атомарных высказываний

В качестве модели системы будем использовать
размеченную систему переходов (СП)[1] $M = (S, S_0, \mapsto, L)$ над AP,
устроенную так:

- $(S, \mapsto, L)$ — это модель Крипке над переменными AP, в которой
  - миры из S называются состояниями,
  - отношение переходов $\mapsto \subseteq S \times S$ тотально: для каждого состояния s существует состояние s', такое что $s \mapsto s'$ — и
  - оценку переменных $L : S \to 2^{\mathsf{AP}}$ принято называть функцией разметки состояний
- S_0 — множество начальных состояний, $S_0 \subseteq S$

СП будем называть конечной,
если конечны множества её состояний и атомарных высказываний

1 На самом деле СП — это более широкое понятие, и модель Крипке с начальными мирами (состояниями) является частным случаем такой системы, но сейчас нет смысла всё переусложнять

Размеченные системы переходов

СП представляет собой размеченный ориентированный граф,
вершинами которого являются состояния, а дугами — переходы,
и поэтому будем использовать для СП терминологию теории графов

Путь в СП будем называть начальным,
если он исходит из начального состояния

Бесконечный начальный путь будем называть вычислением СП

Вычисления СП отвечают (*потенциально*) бесконечным
сценариям выполнения моделируемой системы

Размеченные системы переходов

Пример: СП кофейного автомата

Атомарные высказывания:

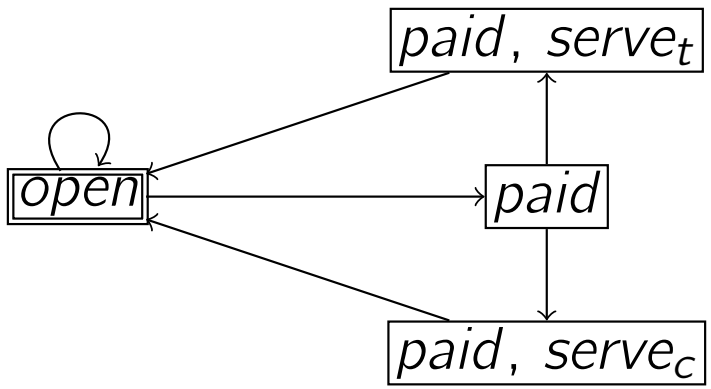
$open$ = «приёмник открыт»

$paid$ = «в приёмнике есть монета»

$serve_t$ = «выдаётся чай»

$serve_c$ = «выдаётся кофе»

СП:



□ — состояние

▣ — начальное состояние

Высказывания, размечающие состояние,
записаны внутри этого состояния

Система переходов программы

Математически строгий анализ императивной программы π
(и программ других парадигм в рамках операционной семантики),
как правило, основан на понятиях, которые уже появлялись в лекциях:

- Состояние управления:
  то, какую часть программы осталось выполнить
- Состояние данных: то, как устроены данные,
  преобразуемые программой на каждом шаге
  (например, оценки переменных или запросы)
- Состояние вычисления,
  включающее в себя состояние данных и состояние управления
- Отношение $\rightarrow_\pi$ шага вычисления программы π

Система переходов программы

Программе π отвечает СП $(S, S_0, \mapsto, L)$ над АР, где:

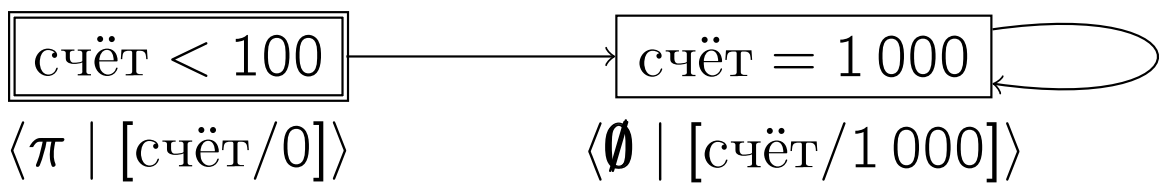
- S — множество всех состояний вычисления π
- S_0 — состояния, с которых начинаются вычисления π на интересующих входных данных
- $\mapsto = \rightarrow_\pi$ с поправкой на требуемую тотальность: если из состояния s не исходит ни одного перехода, то в $\mapsto$ «насильно» добавляется переход $s \mapsto s$
- АР суть интересующие свойства состояний вычисления, например:
  - «$\mathrm{x} = 2$», «$\mathrm{x} > 2$», «$\mathrm{x} > \mathrm{y}$»
  - «Состояние управления — это пустая команда»
  - «Предикатный символ левой подцели — P»
- $L(s)$ состоит из всех атомарных высказываний, истинных для s согласно естественной трактовке их записи

Система переходов программы

Пример

$$\pi = \text{счёт} := \text{счёт} + 1\,000;$$

Достижимый фрагмент СП этой модельной императивной программы для интерпретации $Ar_{\mathbb{Z}}$, начальной оценки $[\text{счёт}/0]$
и атомарных высказываний «$\text{счёт} < 100$» и «$\text{счёт} = 1\,000$»:



Система переходов программы

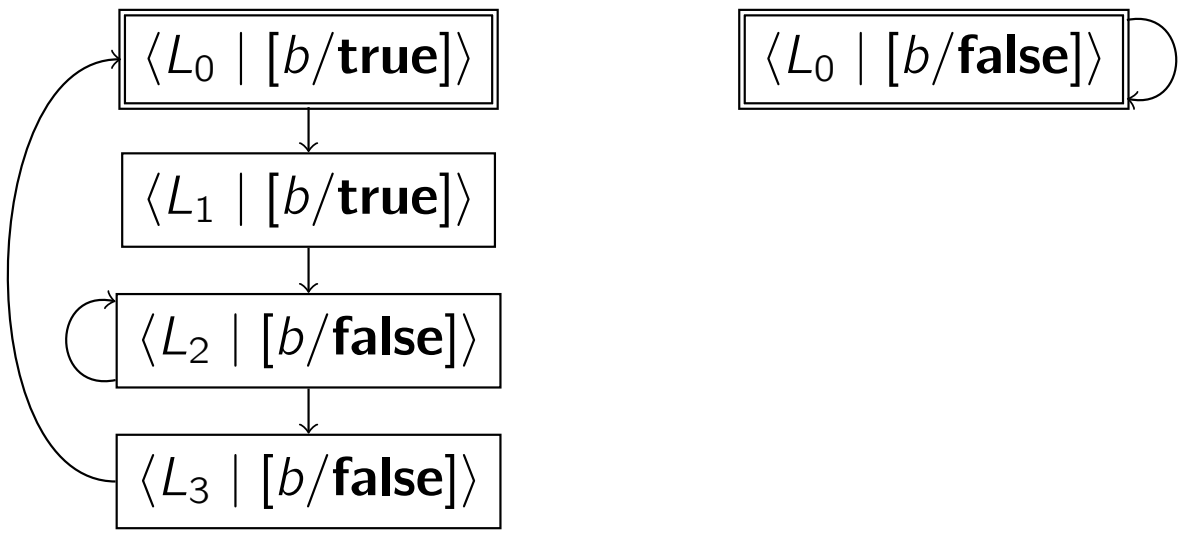
Другой пример

Пусть в сетевом принтере содержится булев регистр $\mathtt{b}$,
обозначающий *занятость* принтера,
и доступ к нему осуществляется при помощи такой программы π:

$$\textbf{while } (\textbf{true}) \{ \quad L_0 : \textbf{while } (!\mathtt{b}); \quad L_1 : \mathtt{b} = \textbf{false}; \quad L_2 : \mathit{EXCHANGE} \quad L_3 : \mathtt{b} = \textbf{true}; \quad \}$$

(*EXCHANGE* — подпрограмма обмена данными для печати)

Фрагмент СП для π, достижимый из начальных состояний
с произвольным начальным значением $\mathtt{b}$, может быть устроен так
(*для простоты считаем, что метка состояния — это оно само*):



Система переходов программы в окружении

Программа в распределённой системе может
взаимодействовать со своим **окружением** при помощи
разделяемых переменных, сообщений, сигналов, ...

Такое взаимодействие выражается в том,
что состояние вычисления программы
может измениться под воздействием окружения

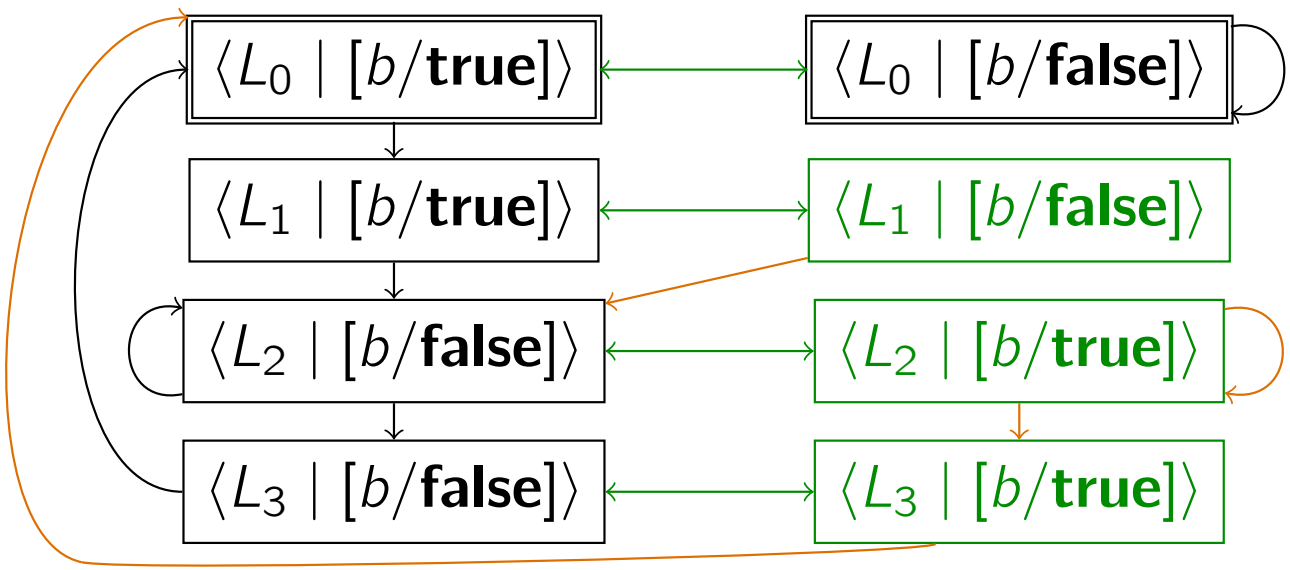
СП программы в окружении — это СП программы,
в которую добавлены переходы, отвечающие возможностям окружения

Система переходов программы в окружении

Пример

$$\textbf{while}\ (\textbf{true})\ \{ \quad L_0 : \textbf{while}\ (!\mathtt{b});\quad L_1 : \mathtt{b} = \textbf{false};$$
$$L_2 : \mathit{EXCHANGE}\quad L_3 : \mathtt{b} = \textbf{true};\quad \}$$

СП для программы доступа к сетевому принтеру
в окружении, способном произвольно изменять значение регистра $\mathtt{b}$,
может быть устроена так:



Параллельная композиция систем переходов

Наиболее популярный способ композиции СП параллельно выполняющихся программ — это асинхронная композиция (согласно семантике чередующихся вычислений; *interleaving*)

Это способ композиции устроен так:

- Состояние композиции компонентов представляет собой набор локальных частей их состояний и *включённую один раз* общую (разделяемую) часть
- Переход в композиции отвечает
  - произвольному выбору одного из компонентов и перехода в нём и
  - выполнению этого перехода с изменением локальной части состояния компонента и общей части состояния согласно устройству СП компонента системы

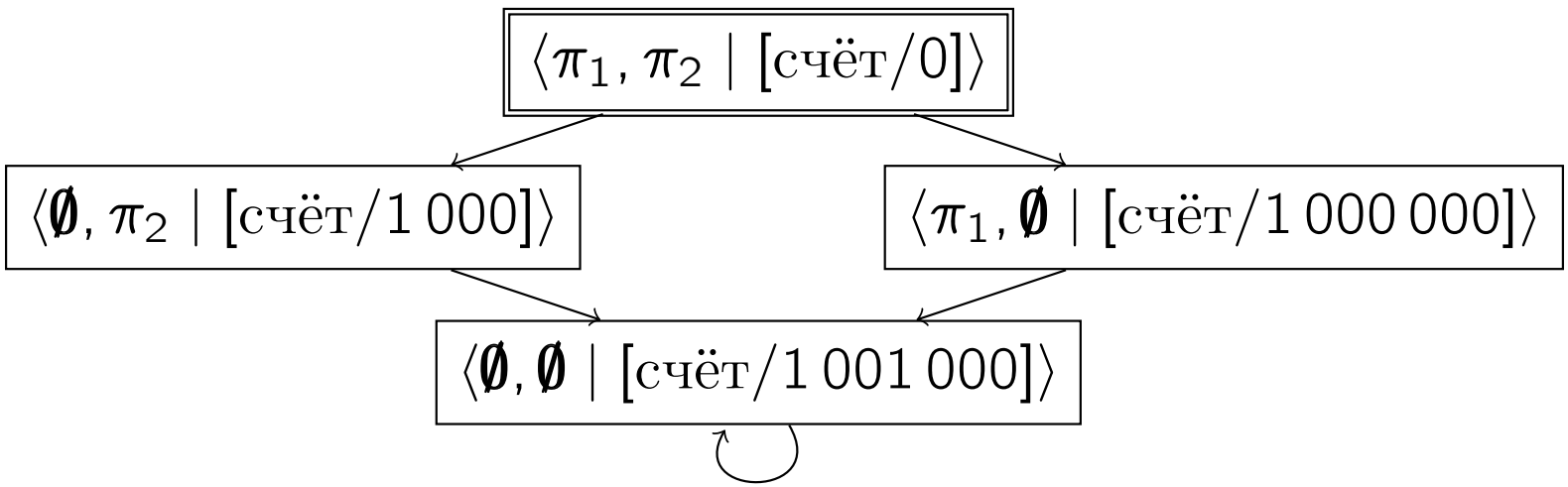
В вычислении построенной так композиции произвольно чередуется выполнение переходов компонентов системы

Параллельная композиция систем переходов

Пример

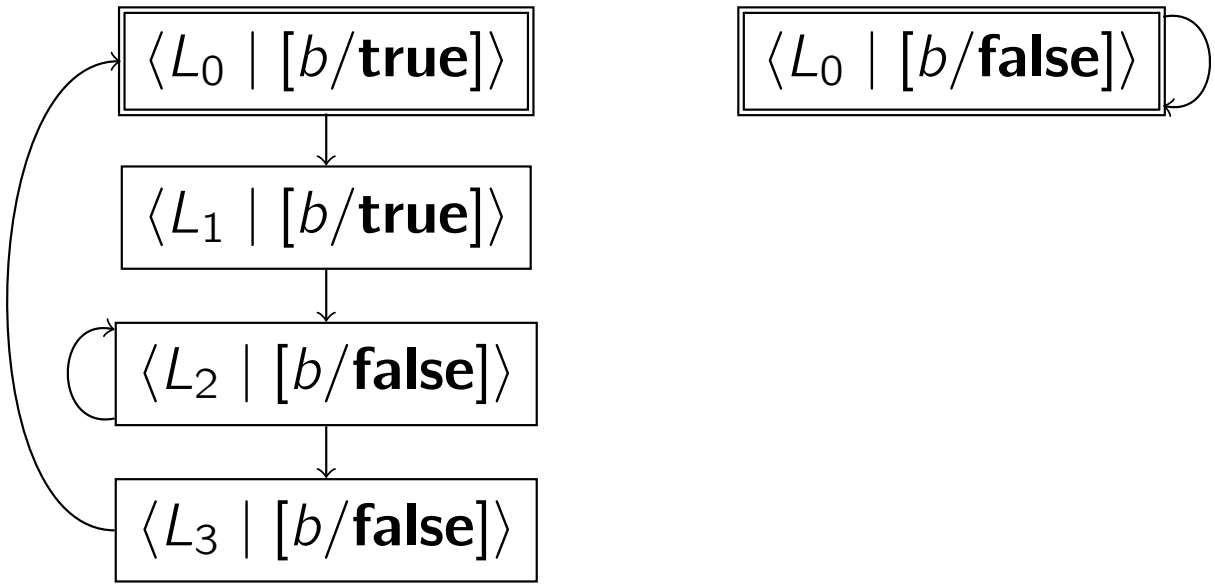
$$\pi_1 = \text{счёт} := \text{счёт} + 1\,000; \qquad \pi_2 = \text{счёт} := \text{счёт} + 1\,000\,000;$$

Достижимый фрагмент асинхронной композиции СП
этих двух модельных императивных программ
с общей переменной «счёт»
для интерпретации $Ar_{\mathbb{Z}}$ и начальной оценки $[\text{счёт}/0]$ устроен так:



Параллельная композиция систем переходов

Другой пример



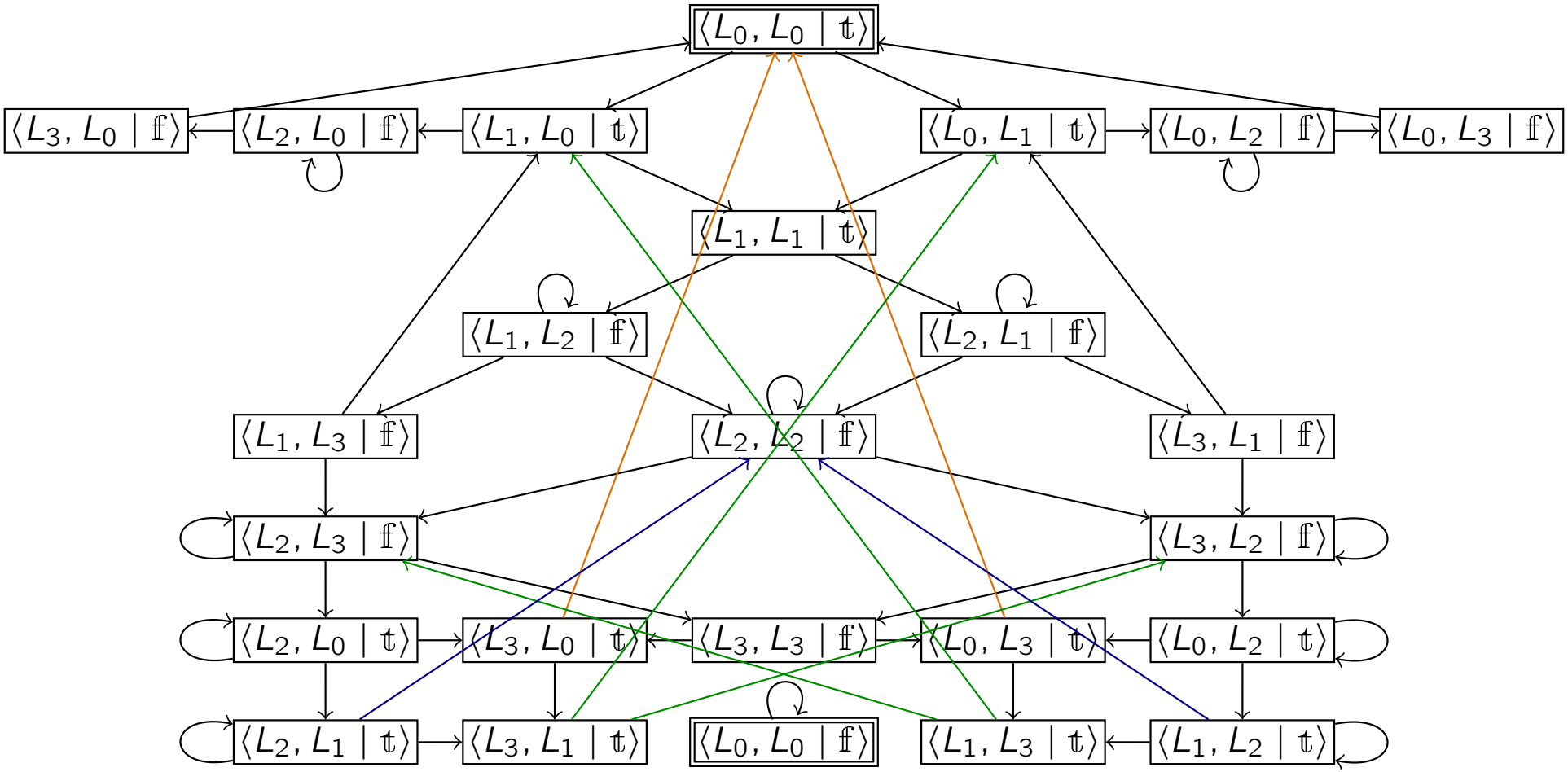
Предположим, что

- с сетевым принтером взаимодействуют две программы с одинаковыми СП (как изображено выше) и
- регистр b является общим для обеих программ

Тогда асинхронная композиция СП, отвечающая параллельному выполнению двух программ доступа к принтеру, устроена так ...

Параллельная композиция систем переходов

Другой пример



Согласно методу проверки моделей,
построение и анализ композиции СП производятся автоматически,
так что «нечитаемость» композиции не считается недостатком

Математическая логика и логическое программирование

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

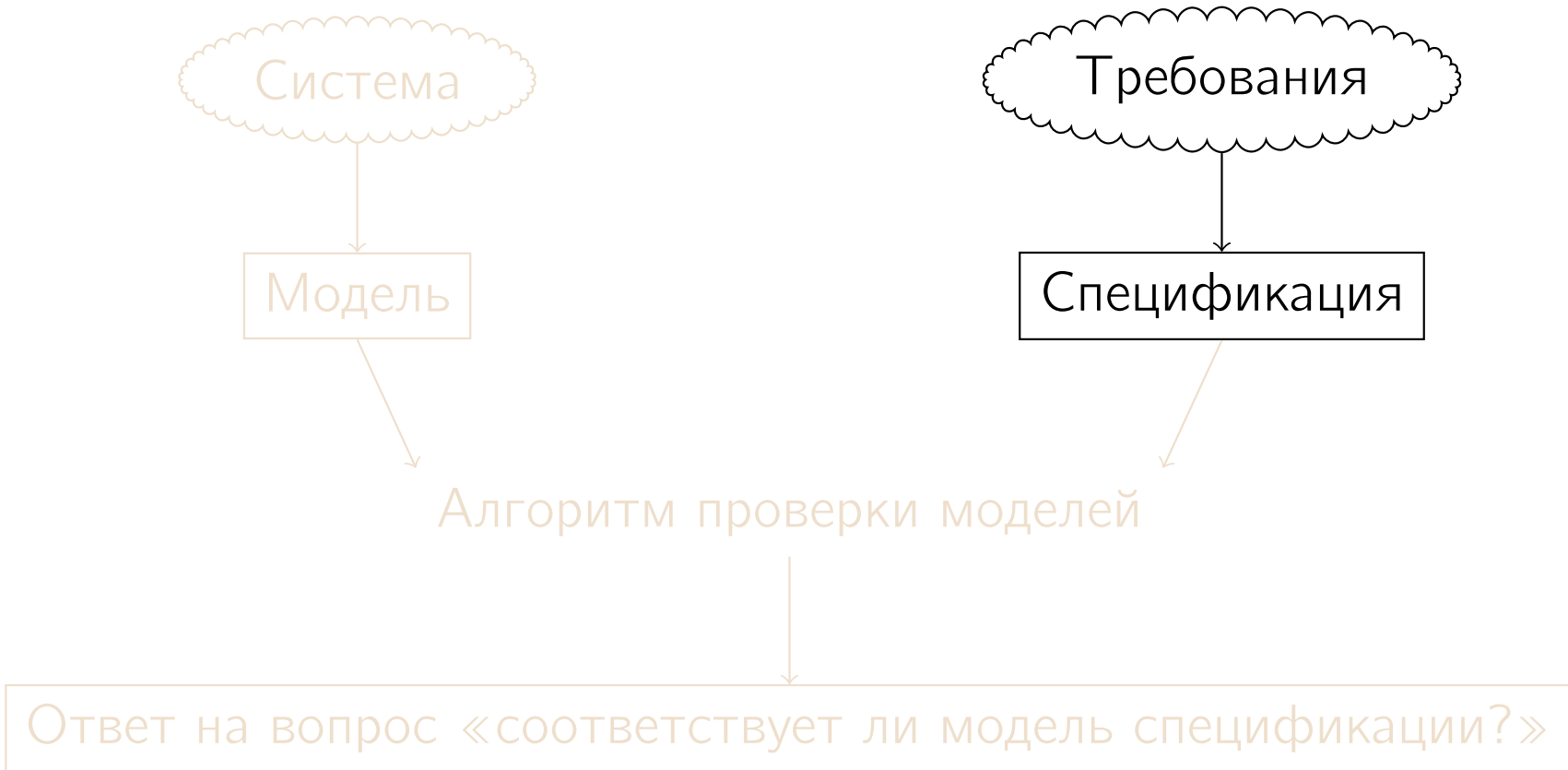
Блок 57

Спецификация систем
при помощи темпоральных логик

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

ВМК МГУ, 2023/2024, осенний семестр

Вступление


Система
Требования
Модель
Спецификация
Алгоритм проверки моделей
Ответ на вопрос «соответствует ли модель спецификации?»

Вступление

Оказалось, что в качестве основы модели распределённой системы можно выбрать модели Крипке:
интерпретацию формул модальной логики

Тогда естественно возникает вопрос:
а нельзя ли в качестве основы языка спецификаций выбрать язык модальных формул?

При положительном ответе можно будет использовать все факты, относящиеся к модальным формулам и их выполнимости в тех или иных моделях

Вступление

Основные препятствия на пути к использованию модальных формул в качестве спецификаций:

- Техническое: когда язык спецификаций выбран, следует строго, чётко и разумно (*адекватно*) поставить задачу проверки соответствия модели и спецификации
- Описательное: если язык спецификаций оказался слишком невыразительным, то требуется найти достаточно выразительное расширение этого языка
- Алгоритмическое: если эффективная проверка соответствия модели и спецификации оказалась невозможной, то требуется найти достаточно эффективно анализируемое сужение этого языка

CTL*

CTL* — это язык спецификаций, который:

- Основан на модальной логике
- Включает в себя LTL и CTL
  - и, в частности, содержит все те буквы (**G**, **F**, **A** и **E**), из которых в блоке 50 строились модальности $\Box$, $\Diamond$ и другие
- Позволяет поставить и решить задачу проверки соответствия модели и формулы
  - То есть задачу проверки выполнимости формулы на СП

CTL*: синтаксис и семантика

Синтаксис формул CTL* над множеством атомарных высказываний AP задаётся БНФ

$$\begin{array}{rcl} \Phi & ::= & \mathbb{t} \mid p \mid (\Phi \,\&\, \Phi) \mid (\Phi \vee \Phi) \mid (\neg\Phi) \mid (\Phi \to \Phi) \\ & \mid & (\mathbf{A}\varphi) \mid (\mathbf{E}\varphi) \\ \varphi & ::= & \Phi \mid (\varphi \,\&\, \varphi) \mid (\varphi \vee \varphi) \mid (\neg\varphi) \mid (\varphi \to \varphi) \\ & \mid & (\mathbf{F}\varphi) \mid (\mathbf{G}\varphi) \mid (\mathbf{X}\varphi) \mid (\varphi \mathbf{U} \varphi), \end{array}$$

где

- Φ — формула CTL*, или, по-другому, формула состояния,
- φ — формула пути и
- $p \in \mathrm{AP}$

Для двух видов формул соответственно определяется два вида выполнимости:

- Выполнимость формулы состояния Φ в заданном состоянии s СП M: $M, s \models \Phi$
- Выполнимость формулы пути φ на заданном бесконечном пути π в СП M: $M, \pi \models \varphi$

CTL*: синтаксис и семантика

Синтаксис формул CTL* над множеством атомарных высказываний AP задаётся БНФ

$$\begin{array}{rcl} \Phi & ::= & \mathbb{t} \mid p \mid (\Phi \,\&\, \Phi) \mid (\Phi \vee \Phi) \mid (\neg\Phi) \mid (\Phi \to \Phi) \\ & \mid & (\mathbf{A}\varphi) \mid (\mathbf{E}\varphi) \\ \varphi & ::= & \Phi \mid (\varphi \,\&\, \varphi) \mid (\varphi \vee \varphi) \mid (\neg\varphi) \mid (\varphi \to \varphi) \\ & \mid & (\mathbf{F}\varphi) \mid (\mathbf{G}\varphi) \mid (\mathbf{X}\varphi) \mid (\varphi \mathbf{U} \varphi), \end{array}$$

Приоритеты операций: $\neg$, **A**, **E**, **F**, **G** и **X**; затем **U**; затем остальные операции с обычными приоритетами

Символ $\mathbb{t}$, связки $\&$, $\vee$, $\neg$, $\to$ и атомарное высказывание p имеют «привычный» содержательный смысл

Буквы **A** и **E** — это кванторы пути:

- «$\mathbf{A}\varphi$» = «для любого бесконечного пути, исходящего из текущего состояния, верно φ» и
- «$\mathbf{E}\varphi$» = «существует бесконечный путь, исходящий из текущего состояния и такой что для него верно φ»

CTL*: синтаксис и семантика

Синтаксис формул CTL* над множеством атомарных высказываний AP задаётся БНФ

$$\begin{array}{rcl} \Phi & ::= & \mathbb{t} \mid p \mid (\Phi \,\&\, \Phi) \mid (\Phi \vee \Phi) \mid (\neg\Phi) \mid (\Phi \to \Phi) \\ & \mid & (\mathbf{A}\varphi) \mid (\mathbf{E}\varphi) \\ \varphi & ::= & \Phi \mid (\varphi \,\&\, \varphi) \mid (\varphi \vee \varphi) \mid (\neg\varphi) \mid (\varphi \to \varphi) \\ & \mid & (\mathbf{F}\varphi) \mid (\mathbf{G}\varphi) \mid (\mathbf{X}\varphi) \mid (\varphi\mathbf{U}\varphi), \end{array}$$

Буквы **F**, **G**, **X**, **U** — это темпоральные операторы:

- «$\mathbf{F}\varphi$» = «когда-нибудь, рано или поздно, станет верно φ»
- «$\mathbf{G}\varphi$» = «всегда будет верно φ»
- «$\mathbf{X}\varphi$» = «в следующем состоянии будет верно φ» (neXt step)
- «$\varphi\mathbf{U}\psi$» = «когда-нибудь станет верно ψ,
  а пока оно не стало верным, обязательно верно φ» (Until)

CTL*: синтаксис и семантика

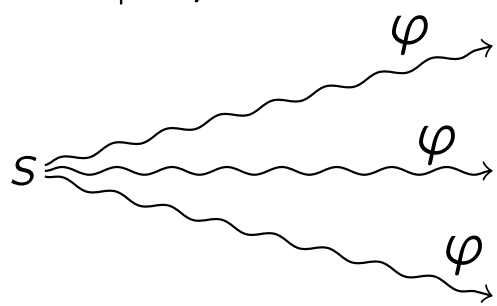
Отношения выполнимости формул для СП $M = (S, S_0, \mapsto, L)$, состояния s и бесконечного пути π задатся следующими правилами:

- Соотношение $M, s \models \mathbb{t}$ верно всегда
- $M, s \models p$, где $p \in \mathrm{AP} \quad \Leftrightarrow \quad p \in L(s)$
- $M, s \models \Phi \,\&\, \Psi \quad \Leftrightarrow \quad M, s \models \Phi$ и $M, s \models \Psi$
- $M, \pi \models \varphi \,\&\, \psi \quad \Leftrightarrow \quad M, \pi \models \varphi$ и $M, \pi \models \psi$
- $M, s \models \Phi \vee \Psi \quad \Leftrightarrow \quad M, s \models \Phi$ или $M, s \models \Psi$
- $M, \pi \models \varphi \vee \psi \quad \Leftrightarrow \quad M, \pi \models \varphi$ или $M, \pi \models \psi$
- $M, s \models \neg\Phi \quad \Leftrightarrow \quad M, s \not\models \Phi$
- $M, \pi \models \neg\varphi \quad \Leftrightarrow \quad M, \pi \not\models \varphi$
- $M, s \models \Phi \to \Psi \quad \Leftrightarrow \quad M, s \not\models \Phi$ или $M, s \models \Psi$
- $M, \pi \models \varphi \to \psi \quad \Leftrightarrow \quad M, \pi \not\models \varphi$ или $M, \pi \models \psi$
- $M, \pi \models \Phi$ для формулы состояния $\Phi \quad \Leftrightarrow \quad M, \pi[1] \models \Phi$
  - $\pi[i]$ — i-е состояние пути π при нумерации с единицы

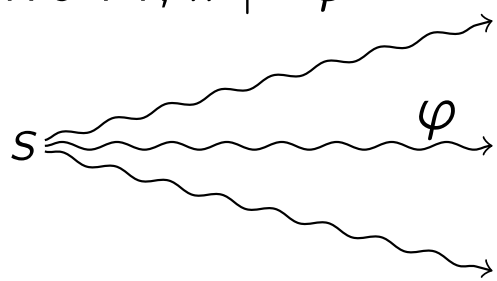
CTL*: синтаксис и семантика

Отношения выполнимости формул для СП $M = (S, S_0, \mapsto, L)$, состояния s и бесконечного пути π задатся следующими правилами:

- $M, s \models \mathbf{A}\varphi \quad \Leftrightarrow \quad$ для любого бесконечного пути π в M, исходящего из s, верно $M, \pi \models \varphi$



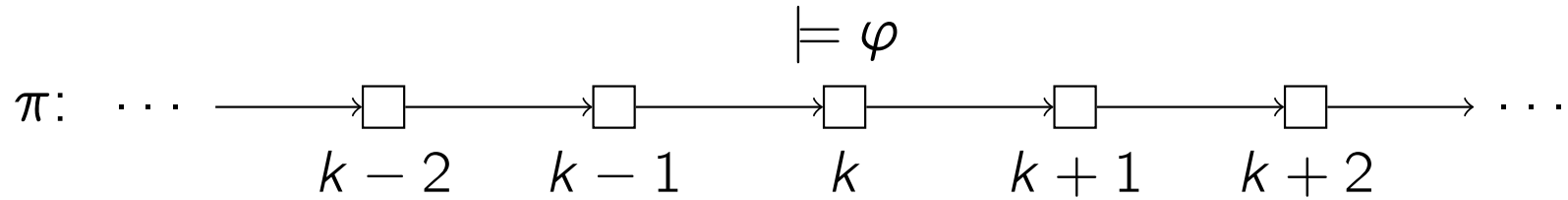
- $M, s \models \mathbf{E}\varphi \quad \Leftrightarrow \quad$ существует бесконечный путь в M, исходящий из s и такой что $M, \pi \models \varphi$



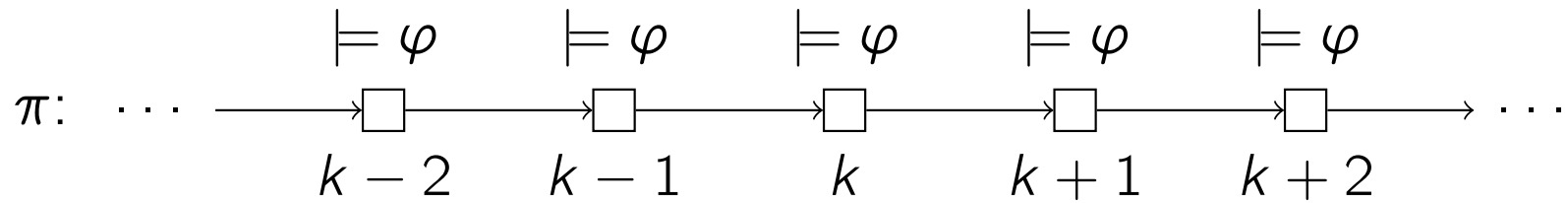
CTL*: синтаксис и семантика

Отношения выполнимости формул для СП $M = (S, S_0, \mapsto, L)$, состояния s и бесконечного пути π задатся следующими правилами:

- $M, \pi \models \mathbf{F}\varphi \quad \Leftrightarrow \quad$ существует номер k, $k \geq 1$, такой что $M, \pi^k \models \varphi$
  - π^k — суффикс пути π, начинающийся с k-го состояния



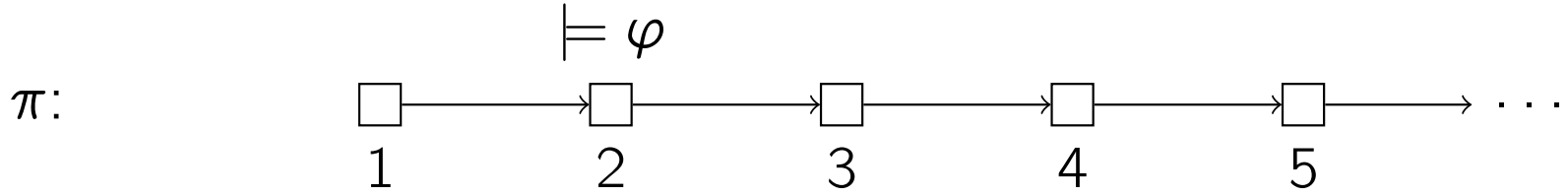
- $M, \pi \models \mathbf{G}\varphi \quad \Leftrightarrow \quad$ для любого номера k, $k \geq 1$, верно $M, \pi^k \models \varphi$



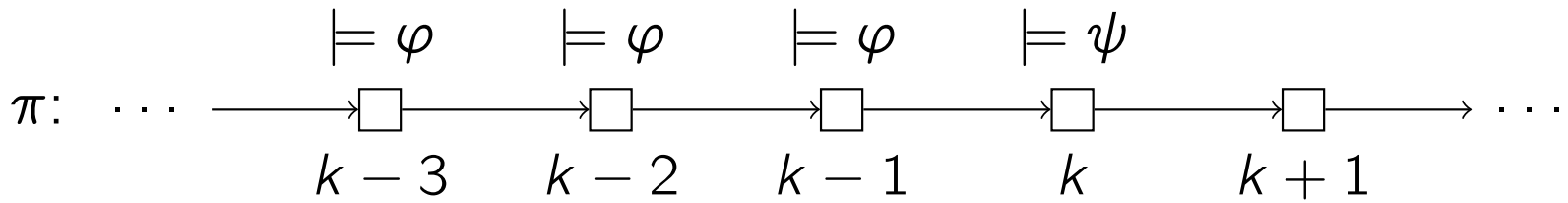
CTL*: синтаксис и семантика

Отношения выполнимости формул для СП $M = (S, S_0, \mapsto, L)$, состояния s и бесконечного пути π задатся следующими правилами:

- $M, \pi \models \mathbf{X}\varphi \quad \Leftrightarrow \quad M, \pi^2 \models \varphi$



- $M, \pi \models \varphi \mathbf{U} \psi \quad \Leftrightarrow \quad$ существует номер k, $k \geq 1$, такой что
  - $M, \pi^k \models \psi$ и
  - для любого номера m, такого что $1 \leq m < k$, верно $M, \pi^m \models \varphi$



CTL*: постановка задачи проверки моделей

Формула CTL* φ выполняется на СП M ($M \models \varphi$),
если она выполняется в любом начальном состоянии системы M

Задача проверки моделей для CTL* формулируется так:
для заданной **конечной** системы переходов M
и заданной формулы φ CTL*
проверить справедливость соотношения $M \models \varphi$

CTL* и LTL

Формула LTL — это формула CTL* вида $\mathbf{A}\varphi$,
в которой подформула φ не содержит ни одного квантора пути
По сравнению с блоком 50, в предложенном определении формулы LTL

- Содержится квантор **A**:
  при использовании формулы LTL в качестве спецификации
  этот квантор принято включать в формулу, явно или неявно
- Содержатся темпоральные операторы **X** и **U**:
  это примеры модальностей, отличных от $\Box$ и $\Diamond$

Бесконечному пути π СП $M = (S, S_0, \mapsto, L)$ можно сопоставить
интерпретацию $\mathcal{I}_\pi = (\mathbb{N}, \leq, \mathcal{L})$ LTL, в которой указаны по порядку
все множества атомарных высказываний, помечающие состояния пути:

$$\mathcal{L}(k) = L(\pi[k])$$

Тогда выполнимость формулы LTL φ на пути π СП совпадает с
её выполнимостью в интерпретации $\mathcal{I}_\pi$ LTL в том смысле,
как это вводилось в рассказе про модальные логики,
и соотношение $M \models \varphi$ означает, что все вычисления M
обладают свойством правильности, записанным в виде формулы φ

CTL* и LTL

Примеры спецификаций на языке LTL:

- Данные на печать передаются не более чем одним принтером:
$$\mathbf{AG}\neg(\mathit{print}_1 \,\&\, \mathit{print}_2)$$
- После завершения процедур начисления стипендии и зарплаты на счёте будет ровно 1 001 000 рублей:
$$\mathbf{AFG}p_{\text{счёт}=1\,001\,000}$$
- На следующем шаге после оплаты напитка он будет выдаваться:
$$\mathbf{AG}(\neg \mathit{paid} \,\&\, \mathbf{X}\mathit{paid} \to \mathbf{XX}(\mathit{serve}_t \vee \mathit{serve}_c))$$
- Если компьютер достаточно часто посылает запрос на печать, то рано или поздно он начнёт печатать:
$$\mathbf{A}(\mathbf{GF}\mathit{request} \to \mathbf{F}\mathit{print})$$
- Если идёт печать, то сеанс печати рано или поздно завершится, и до завершения принтер будет занят:
$$\mathbf{AG}(\mathit{print} \to \mathit{busy}\,\mathbf{U}\neg\mathit{print})$$

CTL* и CTL

Формула CTL — это формула CTL* частного вида, отвечающего БНФ

$$\begin{aligned}\Phi &::= \mathbb{t} \mid p \mid (\Phi \,\&\, \Phi) \mid (\Phi \vee \Phi) \mid (\neg\Phi) \mid (\Phi \to \Phi) \\ &\mid (\mathbf{A}\varphi) \mid (\mathbf{E}\varphi) \\ \varphi &::= (\mathbf{F}\Phi) \mid (\mathbf{G}\Phi) \mid (\mathbf{X}\Phi) \mid (\Phi\mathbf{U}\Phi),\end{aligned}$$

То есть в формуле CTL под квантором пути обязательно располагается темпоральный оператор, и под ним — формула состояния

Иными словами, в формуле CTL кванторы пути и темпоральные операторы используются только в парах:

AG, **EG**, **AF**, **EF**, **AX**, **EX**, **AU**, **EU**

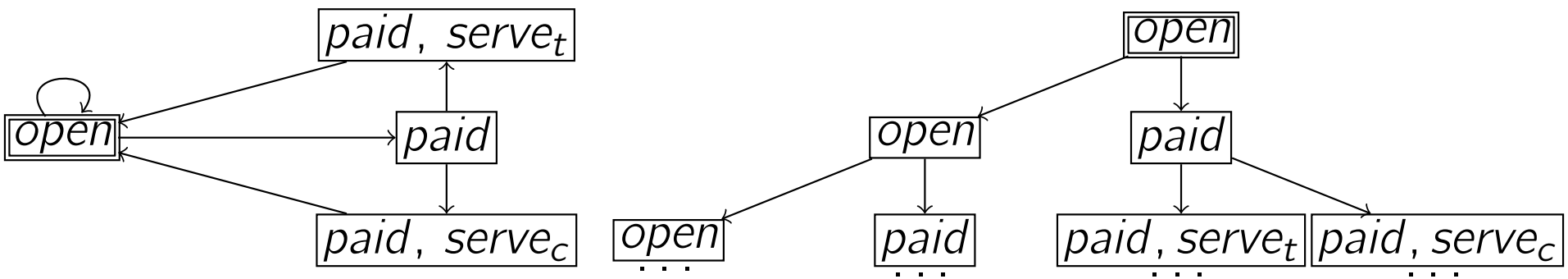
CTL* и CTL

В блоке 50 рассказывалось, что формулы CTL интерпретируются на рефлексивно-транзитивных замыканиях особых бесконечных деревьев

Такое бесконечное дерево можно понимать
как развёртку системы переходов:

- Корень — это выбранное начальное состояние
- Вершина развёртки отвечает конечному пути в СП
  и размечена теми же атомарными высказываниями,
  что и последняя вершина пути
- Дуга $v_1 \to v_2$ в развёртке означает,
  что путь v_1 можно продолжить до пути v_2, добавив один переход

Например, ниже изображены СП и фрагмент её развёртки



CTL* и CTL

Примеры спецификаций на языке CTL для кофейного автомата:

- В самом начале работы автомата приёмник монет открыт, в нём нет монеты, и автомат ничего не выдаёт:
$$open \,\&\, \neg paid \,\&\, \neg serve_t \,\&\, \neg serve_c$$
- Нельзя сделать так, чтобы автомат выдал напиток, не имея монеты в приёмнике:
$$\neg \mathbf{EF}(\neg paid \,\&\, (serve_c \vee serve_t))$$
- Если в приёмнике есть монета, то рано или поздно он выдаст напиток ...
$$\mathbf{AG}(paid \to \mathbf{AF}(serve_c \vee serve_t))$$
- ... но этот напиток не обязан быть чаем ...
$$\mathbf{EF}(paid \,\&\, \mathbf{EG} \neg serve_t)$$
- ... но при желании можно, опустив монету в приёмник, получить чай
$$\mathbf{AG}(\neg paid \to \mathbf{AX}(paid \to \mathbf{EF} serve_t))$$

Математическая логика и логическое программирование

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

Блок 58

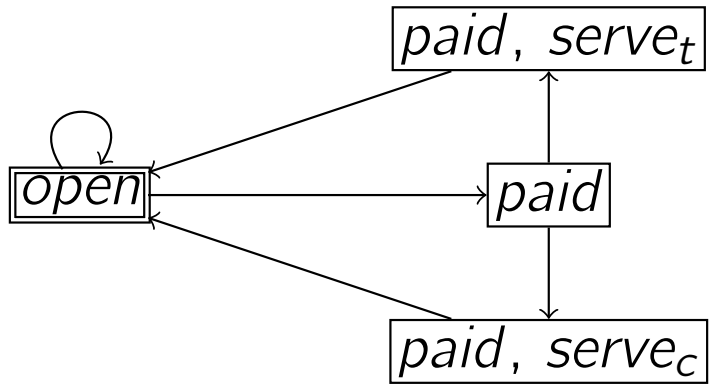
Алгоритм model checking для CTL

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

ВМК МГУ, 2023/2024, осенний семестр

Напоминание

Система переходов M над множеством атомарных высказываний AP:



Примеры формул CTL φ над тем же множеством AP:

$$open \,\&\, \neg paid \,\&\, \neg serve_t \,\&\, \neg serve_c$$
$$\neg \mathbf{EF}(\neg paid \,\&\, (serve_c \vee serve_t))$$
$$\mathbf{AG}(paid \to \mathbf{AF}(serve_c \vee serve_t))$$
$$\mathbf{EF}(paid \,\&\, \mathbf{EG} \neg serve_t)$$
$$\mathbf{AG}(\neg paid \to \mathbf{AX}(paid \to \mathbf{EF} serve_t))$$

$M \models \varphi \Leftrightarrow$
формула φ выполняется в каждом начальном состоянии системы M

Алгоритм проверки моделей для CTL

Алгоритм проверки соотношения $M \models \varphi$ для СП M и формулы φ CTL будет излагаться «сверху вниз» от общей схемы (главной процедуры) к деталям реализации этой схемы (остальным процедурам)

По ходу изложения будет приводиться
обоснование корректности (правильности) каждой процедуры

«Описание алгоритма
+ обоснование корректности
+ оценка сложности» —
типичное сочетание в «умном» изложении алгоритмов, позволяющее

- понять, как это реализовать,
- убедиться, что это действительно работает правильно, и
- оценить, достаточно ли эффективно решение для желаемых целей

Но оценку сложности приводить не будем,
чтобы не перегружать рассказ излишними деталями

Алгоритм проверки моделей для CTL

$Sat(M, \psi)$ — так будем обозначать множество состояний СП M, в которых выполняется формула ψ: $Sat(M, \psi) = \{s \mid s \in S,\ M, s \models \psi\}$

Лемма. **Для любых СП** $M = (S, S_0, \mapsto, L)$ **и формулы** φ **CTL верно:**

$$M \models \varphi \quad \Leftrightarrow \quad S_0 \subseteq Sat(M, \varphi)$$

Доказательство. Напрямую следует из определений соотношения $M \models \varphi$ и множества $Sat(M, \varphi)$ ▼

Главная процедура

Дано: конечная СП M; формула φ CTL

Результат: ответ на вопрос «$M \models \varphi$?»

Тело процедуры:

1. Вычислить множество $X = \Pi_{sat}(M, \varphi) = Sat(M, \varphi)$
2. Проверить соотношение $S_0 \subseteq X$
3. Вернуть результат проверки пункта 2

Алгоритм проверки моделей для CTL

Формулу φ CTL назовём упрощённой, если она задаётся БНФ

$$\varphi \ ::= \ \mathbb{t} \ | \ p \ | \ (\varphi \,\&\, \varphi) \ | \ (\neg\varphi) \ | \ (\mathbf{EX}\varphi) \ | \ (\mathbf{EG}\varphi) \ | \ (\mathbf{E}(\varphi \mathbf{U} \varphi))$$

Формулы ψ_1 и ψ_2 CTL назовём равносильными ($\psi_1 \sim \psi_2$), если для любой СП M верно $Sat(M, \psi_1) = Sat(M, \psi_2)$

Процедура $\Pi_{sat}(M, \varphi)$

Дано: конечная СП M; формула φ CTL

Результат: $Sat(M, \varphi)$

Тело процедуры:

1. Построить упрощённую формулу ψ, равносильную исходной:
$$\psi = Simplify(\varphi)$$
2. Вернуть множество $Sat(M, \psi)$ для упрощённой формулы:
$$\Pi^s_{sat}(M, \psi)$$

Алгоритм проверки моделей для CTL

Лемма (о равносильностях в CTL)
Для любых формул φ и ψ CTL справедливы следующие равносильности:

- $\varphi \to \psi \sim \neg\varphi \vee \psi$
- $\varphi \vee \psi \sim \neg(\neg\varphi \,\&\, \neg\psi)$
- $\mathbf{AX}\varphi \sim \neg\mathbf{EX}\neg\varphi$
- $\mathbf{AF}\varphi \sim \neg\mathbf{EG}\neg\varphi$
- $\mathbf{AG}\varphi \sim \neg\mathbf{EF}\neg\varphi$
- $\mathbf{EF}\varphi \sim \mathbf{E}(\mathbb{t}\mathbf{U}\varphi)$
- $\mathbf{A}(\varphi\mathbf{U}\psi) \sim \neg\mathbf{E}(\neg\psi\mathbf{U}(\neg\varphi \,\&\, \neg\psi)) \,\&\, \neg\mathbf{EG}\neg\psi$

Лемма о равносильностях в CTL

Доказательство. $\varphi ::= \mathbb{t} \mid p \mid \varphi \,\&\, \varphi \mid \neg\varphi \mid \mathbf{EX}\varphi \mid \mathbf{EG}\varphi \mid \mathbf{E}(\varphi \mathbf{U} \varphi)$

$\varphi \to \psi \sim \neg\varphi \vee \psi$ и $\varphi \vee \psi \sim \neg(\neg\varphi \,\&\, \neg\psi)$ — так же как и в логиках высказываний и предикатов

$\mathbf{AX}\varphi \sim \neg\mathbf{EX}\neg\varphi$: покажем, что для любых СП $M = (S, S_0, \mapsto, L)$ и её состояния s верно $\quad M, s \models \mathbf{AX}\varphi \quad \Leftrightarrow \quad M, s \models \neg\mathbf{EX}\neg\varphi$

Верно $M, s \models \mathbf{AX}\varphi$

$\Leftrightarrow$ (по семантике комбинации **AX**)

Для любого состояния s', такого что $s \mapsto s'$, верно $M, s' \models \varphi$

$\Leftrightarrow$ (т.к. $\forall \mathrm{x}\,(A \to B) \sim \neg\exists \mathrm{x}\,(A \,\&\, \neg B)$)

Не существует состояние s', такое что $s \mapsto s'$ и неверно $M, s' \models \varphi$

$\Leftrightarrow$ (по семантике $\neg$)

Не существует состояние s', такое что $s \mapsto s'$ и верно $M, s' \models \neg\varphi$

$\Leftrightarrow$ (по семантике комбинации **EX**)

Неверно $M, s \not\models \mathbf{EX}\neg\varphi$

$\Leftrightarrow$ (по семантике $\neg$)

Верно $M, s \models \neg\mathbf{EX}\neg\varphi$

Лемма о равносильностях в CTL

Доказательство. $\varphi ::= \mathbb{t} \mid p \mid \varphi \,\&\, \varphi \mid \neg\varphi \mid \mathbf{EX}\varphi \mid \mathbf{EG}\varphi \mid \mathbf{E}(\varphi\mathbf{U}\varphi)$

$\mathbf{AF}\varphi \sim \neg\mathbf{EG}\neg\varphi$ и $\mathbf{AG}\varphi \sim \neg\mathbf{EF}\neg\varphi$ — аналогично

$\mathbf{EF}\varphi \sim \mathbf{E}(\mathbb{t}\mathbf{U}\varphi)$ — очевидно следует

из семантики комбинаций **EF** и **EU** и формулы $\mathbb{t}$

$\mathbf{A}(\varphi\mathbf{U}\psi) \sim \neg\mathbf{E}(\neg\psi\mathbf{U}(\neg\varphi \,\&\, \neg\psi)) \,\&\, \neg\mathbf{EG}\neg\psi$:

$M, s \models \mathbf{A}(\varphi\mathbf{U}\psi)$

$\Leftrightarrow$ (по семантике комбинации **AU**)

$\forall$ пути π из s в M $\exists i$: $M, \pi[i] \models \psi$ и $\forall j < i$ верно $M, \pi[j] \models \varphi$

$\Leftrightarrow$ (по двойственности $\forall$-$\exists$ и $\&$-$\vee$)

Не $\exists$ путь π из s в M: $\forall i$ верно ($M, \pi[i] \not\models \psi$ или $\exists j < i$: $M, \pi[j] \not\models \varphi$)

$\Leftrightarrow$ (*применяем метод пристального взгляда*)

1. Не $\exists$ путь π из s в M и номер i:
   $M, \pi[i] \not\models \varphi$, $M, \pi[i] \not\models \psi$ и $\forall j < i$ верно $M, \pi[j] \not\models \psi$
   и
2. не $\exists$ путь π из s в M: $\forall i$ верно $M, \pi[i] \not\models \psi$

$\Leftrightarrow$ (по семантике **E**, **U**, **G**, $\neg$ и $\&$)

$M, s \models \neg\mathbf{E}(\neg\psi\mathbf{U}(\neg\varphi \,\&\, \neg\psi)) \,\&\, \neg\mathbf{EG}\neg\psi$ ▼

Алгоритм проверки моделей для CTL

Процедура *Simplify*(φ)

Дано: формула φ CTL

Результат: упрощённая формула ψ CTL, такая что $\varphi \sim \psi$

Тело процедуры:

1. Пока это возможно, преобразовывать формулу φ согласно равносильностям из последней леммы, заменяя подформулу, отвечающую левой части равносильности, на правую часть
2. Вернуть формулу, получившуюся после всех преобразований

Корректность процедуры *Simplify* обеспечивается тем, что

- наряду с последней леммой для CTL справедлива такая же теорема о равносильной замене, как и для логики предикатов, и
- цикл упрощающих преобразований обязательно завершается: если в исходной формуле содержится n подформул, отвечающих левым частям равносильностей, то после не более чем $2n$ преобразований формула обязательно станет упрощённой, и цикл завершится

Алгоритм проверки моделей для CTL

Процедура $\Pi^s_{sat}(M, \varphi)$

Дано: конечная СП $M = (S, S_0, \mapsto, L)$; упрощённая формула φ CTL

Результат: $Sat(M, \varphi)$

Тело процедуры:

1. Если $\varphi = \mathbb{t}$, то вернуть S
2. Если $\varphi = p \in \mathrm{AP}$, то вернуть $\{s \mid s \in S,\ p \in L(s)\}$
3. Если $\varphi = \psi_1 \,\&\, \psi_2$, то вернуть $\Pi^s_{sat}(M, \psi_1) \cap \Pi^s_{sat}(M, \psi_2)$
4. Если $\varphi = \neg\psi$, то вернуть $S \setminus \Pi^s_{sat}(M, \psi)$
5. Если $\varphi = \mathbf{EX}\psi$, то вернуть $\Pi_{\mathbf{EX}}(M, \psi)$
6. Если $\varphi = \mathbf{EG}\psi$, то вернуть $\Pi_{\mathbf{EG}}(M, \psi)$
7. Если $\varphi = \mathbf{E}(\psi_1 \mathbf{U} \psi_2)$, то вернуть $\Pi_{\mathbf{EU}}(M, \psi_1, \psi_2)$

Корректность этой процедуры для пунктов 1–4 очевидна (обеспечивается семантикой формул)

Осталось предложить подходящие процедуры $\Pi_{\mathbf{EX}}$, $\Pi_{\mathbf{EG}}$ и $\Pi_{\mathbf{EU}}$

Алгоритм проверки моделей для CTL

$Pre(\Gamma, v)$ — так для графа Γ и его вершины v обозначим множество вершин, из которых v достижима по одной дуге:

$$Pre(\Gamma, v) = \{w \mid (w \mapsto v) \in \Gamma\}$$

$Pre(\Gamma, X)$ — так для графа Γ и множества X его вершин обозначим множество вершин, из которых по одной дуге достижима хотя бы одна вершина из X: $Pre(\Gamma, V) = \bigcup\limits_{v \in V} Pre(\Gamma, v)$

Лемма. Для любой СП M и любой формулы φ CTL справедливо равенство $Sat(M, \mathbf{EX}\varphi) = Pre(M, Sat(M, \varphi))$

Доказательство

$s \in Sat(M, \mathbf{EX}\varphi)$ $\Leftrightarrow$ (по определению Sat)

$M, s \models \mathbf{EX}\varphi$ $\Leftrightarrow$ (по семантике **E** и **X**)

$\exists$ состояние s': $s \to s'$ и $M, s' \models \varphi$ $\Leftrightarrow$ (по определению Sat)

$\exists$ состояние множества $Sat(M, \varphi)$, достижимое из s по одной дуге

$\Leftrightarrow$ (по определению Pre)

$s \in Pre(M, Sat(M, \varphi))$ ▼

Алгоритм проверки моделей для CTL

Процедура $\Pi_{\mathbf{EX}}(M, \varphi)$

Дано: конечная СП M; упрощённая формула φ CTL

Результат: $Sat(M, \mathbf{EX}\varphi)$

Тело процедуры:

1. Вычислить $X = \Pi^s_{sat}(M, \varphi)$
2. Вернуть множество $Pre(M, X)$

Алгоритм проверки моделей для CTL

Лемма. **Для любой конечной СП M и любых формул φ_1, φ_2 CTL верно следующее:** $s \in Sat(M, \mathbf{E}(\varphi_1 \mathbf{U} \varphi_2)) \Leftrightarrow$
в M существует путь $s_1 \to \cdots \to s_k$,
такой что $s_1 = s$, $s_k \in Sat(M, \varphi_2)$ и $\{s_1, \ldots, s_{k-1}\} \subseteq Sat(M, \varphi_1)$

Доказательство.

$s \in Sat(M, \mathbf{E}(\varphi_1 \mathbf{U} \varphi_2))$

$\Leftrightarrow$ (по определению Sat)

$M, s \models \mathbf{E}(\varphi_1 \mathbf{U} \varphi_2)$

$\Leftrightarrow$ (по определению **E** и **U**)

$\exists$ бесконечный путь π из s в M и номер k:
$M, \pi[k] \models \varphi_2$ и $\forall i < k$ верно $M, \pi[i] \models \varphi_1$

$\Leftrightarrow$ (переформулировка)

$\exists$ путь $s_1 \mapsto \cdots \mapsto s_k$ в M (префикс пути π):
$s_1 = s$, $M, s_k \models \varphi_2$ и $\forall i \in \{1, \ldots, k-1\}$ верно $M, s_i \models \varphi_1$

$\Leftrightarrow$ (по определению Sat)

$\exists$ путь $s_1 \mapsto \cdots \mapsto s_k$ в M:
$s_1 = s$, $s_k \in Sat(M, \varphi_2)$ и $\{s_1, \ldots, s_{k-1}\} \subseteq Sat(M, \varphi_1)$ ▼

Алгоритм проверки моделей для CTL

Процедура $\Pi_{\mathbf{EU}}(M, \varphi_1, \varphi_2)$

Дано: конечная СП M; упрощённые формулы φ_1, φ_2 CTL

Результат: $Sat(M, \mathbf{E}(\varphi_1 \mathbf{U} \varphi_2))$

Тело процедуры:

1. Вычислить $X_1 = \Pi^s_{sat}(M, \varphi_2)$ и $Z = \Pi^s_{sat}(M, \varphi_1)$
2. Последовательно вычислять множества $X_2, X_3, \ldots$ по схеме $X_i = X_{i-1} \cup (Pre(M, X_{i-1}) \cap Z)$, пока для очередного X_i не окажется верно $X_i = X_{i-1}$
3. Вернуть последнее вычисленное множество X_i

Корректность этой процедуры обосновывается

- последней леммой,
- наблюдением «на грани очевидного» о том, что в множество X_i входят все вершины всех путей вида $s_1 \to \cdots \to s_i$, где $s_i \in Sat(M, \varphi_2)$ и $\{s_1, \ldots, s_{i-1}\} \subseteq Sat(M, \varphi_1)$, и
- гарантированным равенством $X_i = X_{i-1}$ хотя бы для одного i в связи с конечностью M

Алгоритм проверки моделей для CTL

Вершина u достижима из вершины v в ориентированном графе Γ, если в Γ существует путь из v в u (быть может, тривиальный, если $u = v$)

Ориентированный граф сильно связен,
если любые его две вершины достижимы друг из друга

Компонента сильной связности (KCC) ориентированного графа — это максимальный по включению вершин и дуг
сильно связный подграф этого графа

Компонента сильной связности нетривиальна (HKCC),
если в ней содержится хотя бы одна дуга

Алгоритм проверки моделей для CTL

Лемма. В конечном ориентированном графе Γ из вершины s исходит хотя бы один бесконечный путь $\Leftrightarrow$ в Γ из s достижима хотя бы одна НКСС

Доказательство.

($\Leftarrow$) Пусть π — путь из s, оканчивающийся в вершине v НКСС

По выбору v, существует путь из v в v с хотя бы одной дугой

Пусть π' — указанный путь из v в v без первой вершины v

Тогда в Γ содержится и бесконечный путь, исходящий из s:

$$\pi\pi'\pi'\ldots\pi'\ldots$$

($\Rightarrow$) Рассмотрим бесконечный путь π в Γ, исходящий из s

Так как граф Γ конечен, то в π содержится хотя бы одна вершина v, встречающасяся хотя бы два раза: $\pi[i] = \pi[i+k] = v$, $k > 0$

Тогда все вершины множества $\{\pi[i+1], \ldots, \pi[i+k]\}$ достижимы друг из друга, то есть входят в некоторую НКСС,
и эта НКСС достижима из s по пути $\pi[1] \to \cdots \to \pi[i]$ ▼

Алгоритм проверки моделей для CTL

Для ориентированного графа Γ и подмножества V его вершин записью $\Gamma|_V$ обозначим подграф графа Γ, порождённый множеством V:

- Множество вершин $\Gamma|_V$ — это V
- $(s_1, s_2) \in \Gamma|_V \Leftrightarrow \{s_1, s_2\} \subseteq V$ и $(s_1, s_2) \in \Gamma$
- Если граф Γ размечен, то все метки переносятся из Γ в $\Gamma|_V$

Лемма. Для любой конечной модели Крипке M и любой формулы φ CTL верно следующее: $s \in Sat(M, \mathbf{EG}\varphi) \Leftrightarrow s \in M|_{Sat(M,\varphi)}$ **и из s в $M|_{Sat(M,\varphi)}$ достижима хотя бы одна НКСС**

Доказательство.

$s \in Sat(M, \mathbf{EG}\varphi) \Leftrightarrow M, s \models \mathbf{EG}\varphi \Leftrightarrow$

в M существует бесконечный путь π, исходящий из s и такой что $M, \pi[i] \models \varphi$ для каждого момента времени $i \Leftrightarrow$

в $\Gamma = M|_{Sat(M,\varphi)}$ существует бесконечный путь, исходящий из $s \Leftrightarrow$

в Γ содержится s и из неё достижима хотя бы одна НКСС ▼

Алгоритм проверки моделей для CTL

Процедура $\Pi_{\mathbf{EG}}(M, \varphi)$

Дано: конечная СП M; упрощённая формула φ CTL

Результат: $Sat(M, \mathbf{EG}\varphi)$

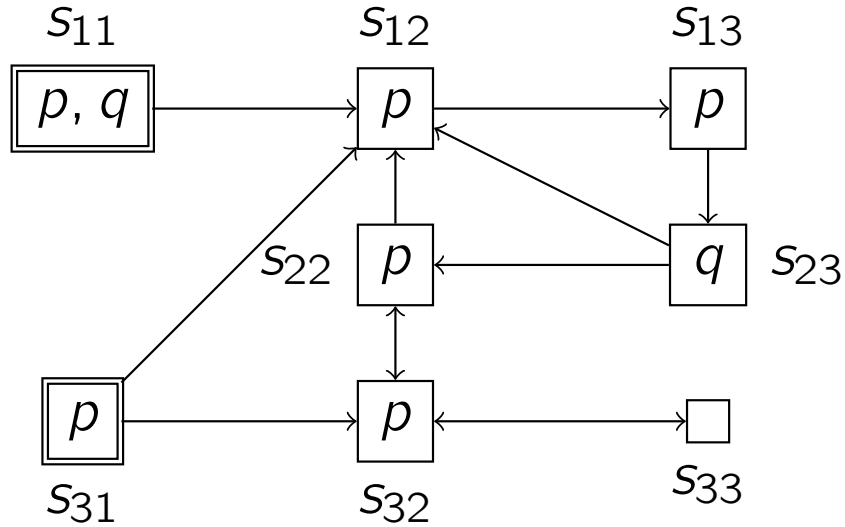
Тело процедуры:

- Вычислить множество $Z = Sat(M, \varphi)$
- Вычислить граф $\Gamma = M|_Z$
- Каким-либо известным эффективным алгоритмом вычислить множество X_1 всех вершин, входящих в какие-либо НКСС в Γ
- Последовательно вычислять множества $X_2, X_3, \ldots$ по схеме $X_i = X_{i-1} \cup Pre(\Gamma, X_{i-1})$, пока для очередного X_i не окажется верно $X_i = X_{i-1}$
- Вернуть последнее вычисленное множество X_i

Корректность этой процедуры обосновывается аналогично корректности $\Pi_{\mathbf{EU}}$

Алгоритм проверки моделей для CTL (пример)

M:



$\varphi = \mathbf{AXA}(p\mathbf{U}q)$

$M \models \varphi?$

$\psi = Simplify(\varphi) = \neg\mathbf{EX}\neg(\neg\mathbf{E}(\neg q\mathbf{U}(\neg q \,\&\, \neg p)) \,\&\, \neg\mathbf{EG}\neg q)$

$\Pi^s_{sat}(M, q) = \{s_{11}, s_{23}\}$

$S = \{s_{11}, s_{12}, s_{13}, s_{22}, s_{23}, s_{31}, s_{32}, s_{33}\}$

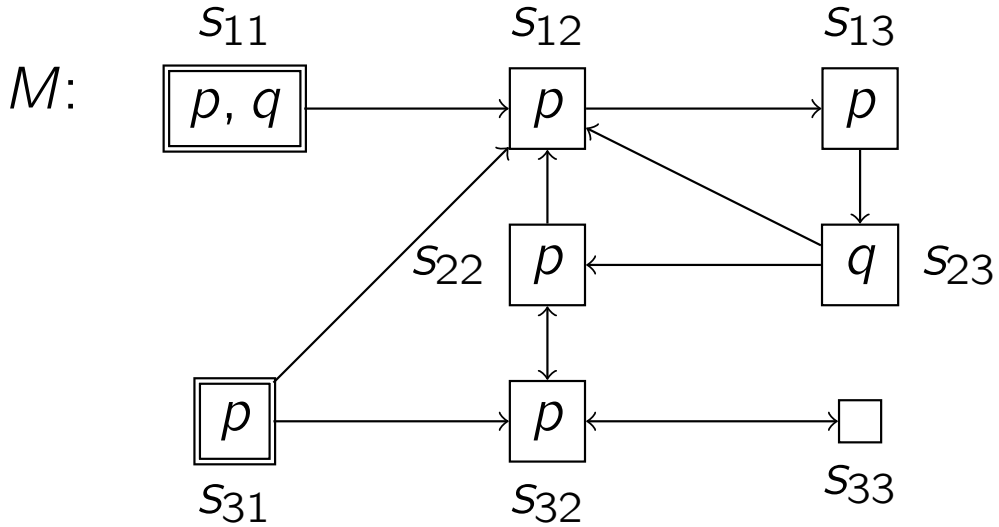
$\Pi^s_{sat}(M, \neg q) = S \setminus \Pi^s_{sat}(M, q) = \{s_{12}, s_{13}, s_{22}, s_{31}, s_{32}, s_{33}\}$

$\Pi^s_{sat}(M, p) = \{s_{11}, s_{12}, s_{13}, s_{22}, s_{31}, s_{32}\}$

$\Pi^s_{sat}(M, \neg p) = S \setminus \Pi^s_{sat}(M, p) = \{s_{23}, s_{33}\}$

$\Pi^s_{sat}(M, \neg q \,\&\, \neg p) = \Pi^s_{sat}(M, \neg q) \cap \Pi^s_{sat}(M, \neg p) = \{s_{33}\}$

Алгоритм проверки моделей для CTL (пример)



$\varphi = \mathbf{AXA}(p\mathbf{U}q)$

$M \models \varphi$?

$\psi = Simplify(\varphi) = \neg\mathbf{EX}\neg(\neg\mathbf{E}(\underbrace{\neg q}_{\chi_1}\ \mathbf{U}\ \underbrace{(\neg q\ \&\ \neg p)}_{\chi_2})\ \&\ \neg\mathbf{EG}\neg q)$

$\Pi^s_{sat}(M, \chi_1) = \{s_{12}, s_{13}, s_{22}, s_{31}, s_{32}, s_{33}\}$

$\Pi^s_{sat}(M, \chi_2) = \{s_{33}\}$

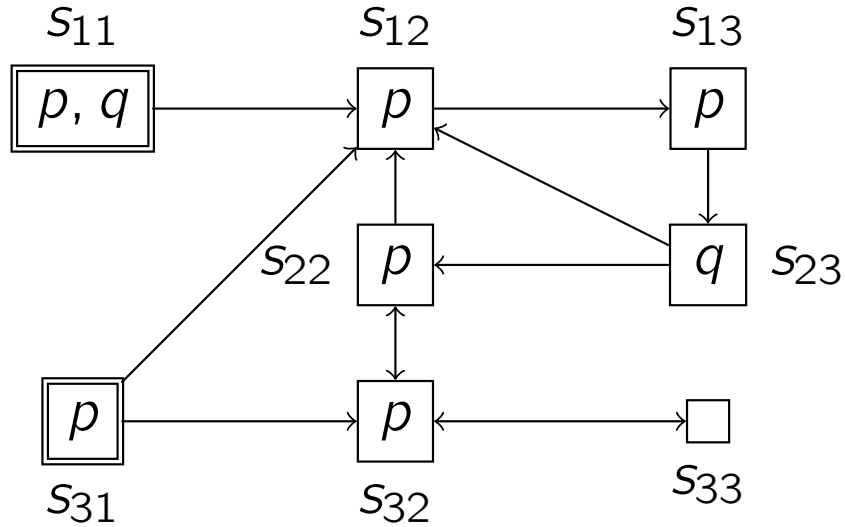
$\Pi^s_{sat}(M, \mathbf{E}(\chi_1\mathbf{U}\chi_2)) =?$

- $X_1 = \Pi^s_{sat}(M, \chi_2),\ Z = \Pi^s_{sat}(M, \chi_1)$
- $X_2 = X_1 \cup (Pre(M, X_1) \cap Z) = \{s_{32}, s_{33}\}$
- $X_3 = X_2 \cup (Pre(M, X_2) \cap Z) = \{s_{22}, s_{31}, s_{32}, s_{33}\}$
- $X_4 = X_3 \cup (Pre(M, X_3) \cap Z) = \{s_{22}, s_{31}, s_{32}, s_{33}\} = X_3$

$\Pi^s_{sat}(M, \mathbf{E}(\chi_1\mathbf{U}\chi_2)) = X_4 = \{s_{22}, s_{31}, s_{32}, s_{33}\}$

Алгоритм проверки моделей для CTL (пример)

M:



$\varphi = \mathbf{AXA}(p\mathbf{U}q)$

$M \models \varphi$?

$\psi = Simplify(\varphi) = \neg\mathbf{EX}\neg(\neg\mathbf{E}(\neg q\mathbf{U}(\neg q \,\&\, \neg p)) \,\&\, \neg\mathbf{EG}\underbrace{\neg q}_{\chi})$

$\Pi^s_{sat}(M, \chi) = \{s_{12}, s_{13}, s_{22}, s_{31}, s_{32}, s_{33}\}$

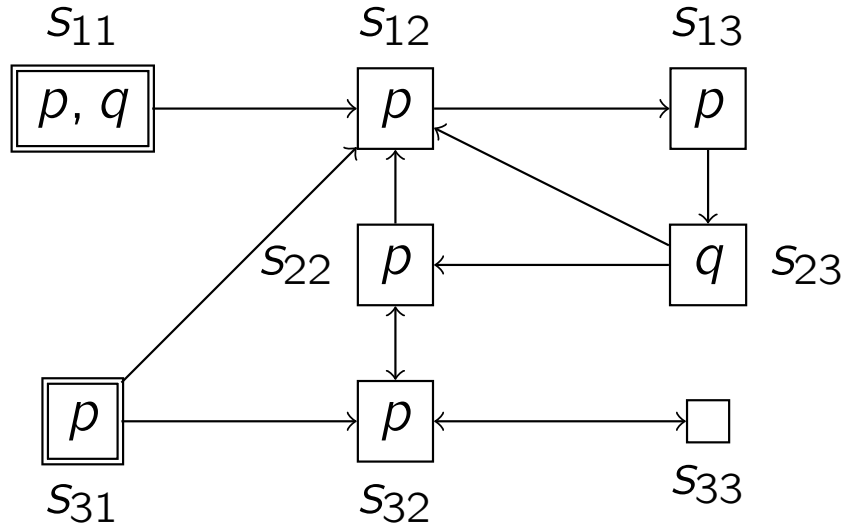
$\Pi^s_{sat}(M, \mathbf{EG}\chi) =$?

- $Z = \Pi^s_{sat}(M, \chi)$
- В графе $M|_Z$ содержится ровно одна нетривиальная компонента сильной связности, и её вершины: $X_1 = \{s_{22}, s_{32}, s_{33}\}$
- $X_2 = X_1 \cup Pre(M|_Z, X_1) = \{s_{22}, s_{31}, s_{32}, s_{33}\}$
- $X_3 = X_2 \cup Pre(M|_Z, X_2) = \{s_{22}, s_{31}, s_{32}, s_{33}\} = X_2$

$\Pi^s_{sat}(M, \mathbf{EG}\chi) = X_3 = \{s_{22}, s_{31}, s_{32}, s_{33}\}$

Алгоритм проверки моделей для CTL (пример)

M:



$\varphi = \mathbf{AXA}(p\mathbf{U}q)$

$M \models \varphi$?

$$\psi = Simplify(\varphi) = \neg\mathbf{EX}\neg(\neg\underbrace{\mathbf{E}(\neg q\mathbf{U}(\neg q \,\&\, \neg p))}_{\chi_1} \,\&\, \neg\underbrace{\mathbf{EG}\neg q}_{\chi_2})$$

$$S = \{s_{11}, s_{12}, s_{13}, s_{22}, s_{23}, s_{31}, s_{32}, s_{33}\}$$

$$\Pi^s_{sat}(M, \chi_1) = \{s_{22}, s_{31}, s_{32}, s_{33}\}$$

$$\Pi^s_{sat}(M, \chi_2) = \{s_{22}, s_{31}, s_{32}, s_{33}\}$$

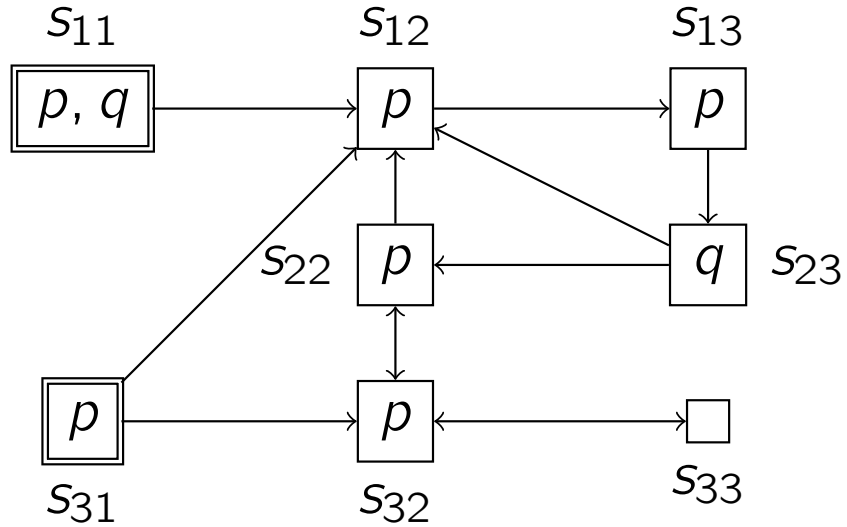
$$\Pi^s_{sat}(M, \neg\chi_1) = S \setminus \Pi^s_{sat}(M, \chi_1) = \{s_{11}, s_{12}, s_{13}, s_{23}\}$$

$$\Pi^s_{sat}(M, \neg\chi_2) = S \setminus \Pi^s_{sat}(M, \chi_1) = \{s_{11}, s_{12}, s_{13}, s_{23}\}$$

$$\Pi^s_{sat}(M, \neg\chi_1 \,\&\, \neg\chi_2) = \Pi^s_{sat}(M, \chi_1) \cap \Pi^s_{sat}(M, \chi_2) = \{s_{11}, s_{12}, s_{13}, s_{23}\}$$

Алгоритм проверки моделей для CTL (пример)

M:



$\varphi = \mathbf{AXA}(p\mathbf{U}q)$

$M \models \varphi$?

$$\psi = Simplify(\varphi) = \neg\mathbf{EX}\neg\underbrace{(\neg\mathbf{E}(\neg q\mathbf{U}(\neg q \,\&\, \neg p)) \,\&\, \neg\mathbf{EG}\neg q)}_{\chi}$$

$S = \{s_{11}, s_{12}, s_{13}, s_{22}, s_{23}, s_{31}, s_{32}, s_{33}\}$

$\Pi^s_{sat}(M, \chi) = \{s_{11}, s_{12}, s_{13}, s_{23}\}$

$\Pi^s_{sat}(M, \neg\chi) = S \setminus \Pi^s_{sat}(M, \chi) = \{s_{22}, s_{31}, s_{32}, s_{33}\}$

$\Pi^s_{sat}(M, \mathbf{EX}\neg\chi) = Pre(M, \Pi^s_{sat}(M, \neg\chi)) = \{s_{22}, s_{23}, s_{31}, s_{32}, s_{33}\}$

$\Pi^s_{sat}(M, \psi) = S \setminus \Pi^s_{sat}(M, \mathbf{EX}\neg\chi) = \{s_{11}, s_{12}, s_{13}\}$

$S_0 = \{s_{11}, s_{31}\} \not\subseteq \Pi^s_{sat}(M, \psi)$

Следовательно, $M \not\models \varphi$