Математические методы верификации схем и программ

mk.cs.msu.ru → Лекционные курсы
→ Математические методы верификации схем и программ

Блок 3

Общие принципы дедуктивной верификации программ

Модельные императивные программы:
синтаксис,
операционная семантика

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

ВМК МГУ, 2024/2025, осенний семестр

Принципы дедуктивной верификации программ

1. Программа π вычисляет отношение R_π между данными, которые подаются на вход и получаются на выходе

2. Текст программы π — это формальное описание отношения R_π

3. Спецификация Φ программы — это формальное описание отношения R_Φ между данными программы
   - Отношение, описываемое спецификацией, — это требования, которым должно удовлетворять отношение, вычисляемое программой

4. Формальная верификация программы π относительно спецификации Φ — это строгое доказательство того, что программа π удовлетворяет требованиям Φ, то есть доказательство включения $R_\pi \subseteq R_\Phi$

Принципы дедуктивной верификации программ

Чтобы уметь формально верифицировать программы, нужно:

1. Строго описать,
   - какие записи мы считаем программами (синтаксис программ)
   - как программы преобразуют входные данные в выходные (семантику программ)
2. Выбрать формальный язык описания требований к программам
3. Предложить метод проверки того, удовлетворяет ли заданная программа предъявленным к ней требованиям

Синтаксис программ

Синтаксис императивных программ (заданной сигнатуры σ) зададим следующей БНФ:

$$
\begin{array}{lcll}
\pi & ::= & stmt \ \mid \ stmt\ \pi & \\
stmt & ::= & \emptyset \ \mid & (\text{пустая команда}) \\
 & & \mathrm{x} := t; \ \mid & (\text{присваивание}) \\
 & & \textbf{if } C \textbf{ then } \pi \textbf{ else } \pi \textbf{ fi} \ \mid & (\text{ветвление}) \\
 & & \textbf{while } C \textbf{ do } \pi \textbf{ od} & (\text{цикл})
\end{array}
$$

Здесь:

- π — программа
- $stmt$ — команда программы (или, по-другому, инструкция)
- $\mathrm{x} \in \mathrm{Var}$
- t — выражение: произвольный терм
- C — условие: произвольная формула без $\forall$ и $\exists$

Синтаксис программ

Пример: реализация алгоритма Эвклида
вычисления наибольшего общего делителя чисел в переменных x, y

```
while ¬(x = y) do
  if x > y then
    x := x − y;
  else
    y := y − x;
  fi
od
```

Виды семантики программ

Два основных способа определения семантики программ:

денотационный и операционный

Денотационный:

- Значение каждой части программы — это денотация (денотат, denotation), особый математический объект
- Денотация программы задаётся как композиция денотаций её составных частей
- Денотационной семантикой не определяется способ вычисления денотации на входных данных
- Хорошо подходит для описания значения функциональных программ
- Денотации функциональных программ — это функции

Виды семантики программ

Два основных способа определения семантики программ:

денотационный и операционный

Операционный:

- Вычисление программы — это последовательное изменение текущего состояния вычисления
- Программой задаётся отношение переходов, описывающее то, какое состояние будет следующим в вычислении относительно произвольного текущего состояния
- Значение программы — это функция преобразования входных данных в выходные, определяемая на основе *рефлексивно-транзитивного замыкания* отношения переходов
- Хорошо подходит для описания значения императивных программ

Операционная семантика программ

Состояние управления — это произвольная программа

Состояние данных — это произвольная оценка всех переменных Var

Состояние вычисления — это пара $\langle \pi \mid \sigma \rangle$, где
π — состояние управления и σ — состояние данных

Σ — так будем обозначать множество всех состояний данных

В примерах будем записывать состояния данных
как оценки только «заслуживающих внимания» переменных,
считая, что значения остальных переменных неважны

Операционная семантика программ

Отношение переходов $\xrightarrow{\mathcal{I}}$ на множестве состояний вычисления в интерпретации $\mathcal{I}$ определяется так:

- $\langle \mathrm{x} := t; \mid \sigma\rangle \xrightarrow{\mathcal{I}} \langle \emptyset \mid \sigma[\mathrm{x} \leftarrow t\sigma]\rangle$

- Если $\mathcal{I} \models C\sigma$, то $\langle \textbf{if } C \textbf{ then } \pi_1 \textbf{ else } \pi_2 \textbf{ fi} \mid \sigma\rangle \xrightarrow{\mathcal{I}} \langle \pi_1 \mid \sigma\rangle$
- Если $\mathcal{I} \not\models C\sigma$, то $\langle \textbf{if } C \textbf{ then } \pi_1 \textbf{ else } \pi_2 \textbf{ fi} \mid \sigma\rangle \xrightarrow{\mathcal{I}} \langle \pi_2 \mid \sigma\rangle$

- Если $\mathcal{I} \models C\sigma$, то $\langle \textbf{while } C \textbf{ do } \pi \textbf{ od} \mid \sigma\rangle \xrightarrow{\mathcal{I}} \langle \pi \textbf{ while } C \textbf{ do } \pi \textbf{ od} \mid \sigma\rangle$
- Если $\mathcal{I} \not\models C\sigma$, то $\langle \textbf{while } C \textbf{ do } \pi \textbf{ od} \mid \sigma\rangle \xrightarrow{\mathcal{I}} \langle \emptyset \mid \sigma\rangle$

- Если $\langle \pi_1 \mid \sigma\rangle \xrightarrow{\mathcal{I}} \langle \pi_1' \mid \sigma'\rangle$, то $\langle \pi_1\ \pi_2 \mid \sigma\rangle \xrightarrow{\mathcal{I}} \langle \pi_1'\ \pi_2 \mid \sigma'\rangle$

- $\langle \emptyset\ \pi \mid \sigma\rangle \xrightarrow{\mathcal{I}} \langle \pi \mid \sigma\rangle$

Операционная семантика программ

Трасса программы π на оценке σ в интерпретации $\mathcal{I}$ — это последовательность состояний вычисления вида

$$\langle \pi \mid \sigma \rangle \xrightarrow{\mathcal{I}} \langle \pi_1 \mid \sigma_1 \rangle \xrightarrow{\mathcal{I}} \langle \pi_2 \mid \sigma_2 \rangle \xrightarrow{\mathcal{I}} \dots$$

Вычисление программы π на оценке σ в интерпретации $\mathcal{I}$ — это максимальная по длине трасса π на σ в $\mathcal{I}$

Результат конечного вычисления программы π
на оценке σ в интерпретации $\mathcal{I}$ — это
оценка в последнем состоянии вычисления π на σ в $\mathcal{I}$

Все бесконечные вычисления считаются не имеющими результата

Операционная семантика программ

Рефлексивно-транзитивное замыкание отношения $R \subseteq X \times X$ — это наименьшее по включению отношение $R^* \subseteq X \times X$, удовлетворяющее следующим свойствам:

- $R \subseteq R^*$
- $\{(x, x) \mid x \in X\} \subseteq R^*$
- Если $(x, y) \in R^*$ и $(y, z) \in R^*$, то $(x, z) \in R^*$

Утверждение. **Для любых программы π, интерпретации $\mathcal{I}$ и состояний данных σ, $\overline{\sigma}$ верно следующее:**

$$\langle \pi \mid \sigma \rangle \xrightarrow{\mathcal{I}}{}^{*} \langle \emptyset \mid \overline{\sigma} \rangle \quad \Leftrightarrow \quad \overline{\sigma} \text{ — результат вычисления } \pi \text{ в } \mathcal{I} \text{ на } \sigma$$

Программой π в интерпретации $\mathcal{I}$ реализуется отношение $R_\pi^\mathcal{I} \subseteq \Sigma \times \Sigma$, задающееся так:

$$(\sigma, \overline{\sigma}) \in R_\pi^\mathcal{I} \quad \Leftrightarrow \quad \langle \pi \mid \sigma \rangle \xrightarrow{\mathcal{I}}{}^{*} \langle \emptyset \mid \overline{\sigma} \rangle$$

Пример вычисления программы

$$\pi = \textbf{while } \neg(x = y) \textbf{ do if } x > y \textbf{ then } x := x - y; \textbf{ else } y := y - x; \textbf{ fi od}$$

Вычисление π на $[x/2, y/4]$ в $\mathcal{I}_{ar}$:

$$\begin{array}{ll}
\langle \pi \mid [x/2, y/4] \rangle & \\
\quad \mathcal{I}_{ar} \downarrow & \text{т.к. } \mathcal{I}_{ar} \models \neg(x = y)[x/2, y/4] \\
\langle \textbf{if } x > y \textbf{ then } x := x - y; \textbf{ else } y := y - x; \textbf{ fi } \pi \mid [x/2, y/4] \rangle & \\
\quad \mathcal{I}_{ar} \downarrow & \text{т.к. } \mathcal{I}_{ar} \not\models (x > y)[x/2, y/4] \\
\langle y := y - x; \pi \mid [x/2, y/4] \rangle & \\
\quad \mathcal{I}_{ar} \downarrow & \text{т.к. } [x/2, y/4][y \leftarrow (y - x)[x/2, y/4]] = [x/2, y/2] \\
\langle \emptyset\, \pi \mid [x/2, y/2] \rangle & \\
\quad \mathcal{I}_{ar} \downarrow & \\
\langle \pi \mid [x/2, y/2] \rangle & \\
\quad \mathcal{I}_{ar} \downarrow & \text{т.к. } \mathcal{I}_{ar} \not\models \neg(x = y)[x/2, y/2] \\
\langle \emptyset \mid [x/2, y/2] \rangle &
\end{array}$$

Результат этого вычисления: $[x/2, y/2]$

Следовательно, $([x/2, y/4], [x/2, y/2]) \in R_{\pi}^{\mathcal{I}_{ar}}$