Математические модели последовательных вычислений

mk.cs.msu.ru → Лекционные курсы
→ Математические модели последовательных вычислений

Блок 13

Схемы Ляпунова-Янова

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

ВМК МГУ, 2022/2023, весенний семестр

Схема Ляпунова-Янова строится над двумя заранее заданными конечными множествами (алфавитами):

- операторных символов
- логических символов

Операторные символы отвечают командам преобразования данных

- (как, например, команда присваивания)

Логические символы отвечают условиям ветвления: булевым выражениям, в зависимости от принимаемого значения передающим управления

- (как, например, команда ветвления **if** – **then** – **else**)

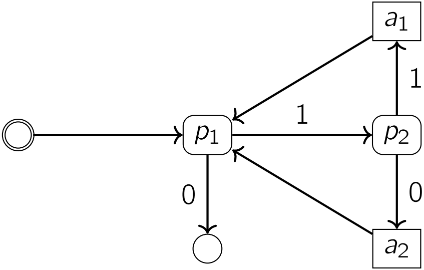
Схема Ляпунова-Янова над алфавитами операторых символов $\mathcal{A}$ и логических символов $\mathcal{P}$ — это конечный ориентированный размеченный граф, все вершины которого разделены на 4 класса:

1. Вход ($\circledcirc$)
   Из него исходит ровно одна дуга
   В него не входит ни одной дуги
   В схеме содержится ровно один вход
2. Выход ($\bigcirc$)
   Из него не исходит ни одной дуги
   В схеме содержится ровно один выход
3. Преобразователь ($\boxed{a}$)
   Он помечен символом из $\mathcal{A}$
   Из него исходит ровно одна дуга
4. Распознаватель (p)
   Он помечен символом из $\mathcal{P}$
   Из него исходит ровно две дуги, одна помечена символом 0, другая — символом 1

Пример

```
while (x != y)
  if (x > y)
    x = x - y;
  else
    y = y - x;
```

Если p_1 = «x != y», p_2 = «x > y»,
a_1 = «x = x - y» и a_2 = «y = y - x»,
то этой программе соответствует такая схема Ляпунова-Янова:



Чтобы придать смысл операторным и логическим символам, следует задать

- множество состояний данных, над которыми выполняется схема,
- способ преобразования состояний данных операторными символами и
- значение каждого логического символа в каждом состоянии данных

Состояниями данных схемы являются слова в алфавите $\mathcal{A}$ (конечные последовательности элементов $\mathcal{A}$)

Иными словами, про состояние данных ничего не известно, кроме того, выполнение какой последовательности операторов привело в это состояние, и преобразование такого состояния — это дописывание соответствующего операторного символа в конец

A^* — так будем обозначать множество всех слов в алфавите $\mathcal{A}$

Значение каждого логического символа на каждом слове из операторных символов задаётся функцией разметки $\mu : \mathcal{A}^* \times \mathcal{P} \to \{0, 1\}$

Состояние вычисления схемы Σ — это пара (v, α), где v — вершина схемы и $\alpha \in \mathcal{A}^*$

Вычисление схемы π на функции разметки μ — это (конечная или бесконечная) последовательность состояний вычисления

$$(v_1, \alpha_1), (v_2, \alpha_2), (v_3, \alpha_3), \dots,$$

устроенная так:

1. $v_1 = \circledcirc$, $\alpha_1 = \lambda$, $v_1 \to v_2$ и $\alpha_2 = \lambda$
2. Если $v_i = \boxed{\mathrm{a}}$, то $v_i \to v_{i+1}$ и $\alpha_{i+1} = \alpha_i a$
3. Если $v_i = \boxed{p}$, то $v_i \xrightarrow{\mu(\alpha_i, p)} v_{i+1}$ и $\alpha_{i+1} = \alpha_i$
4. Если $v_i = \bigcirc$, то (v_i, α_i) — последняя вершина последовательности

Результат $\overline{\pi}(\mu)$ вычисления схемы π на функции разметки μ определяется так:

1. Если вычисление π на μ конечно и оканчивается парой (v_i, α_i), то $\overline{\pi}(\mu) = \alpha_i$
2. Если вычисление π на μ бесконечно, то $\overline{\pi}(\mu) = \perp$, где $\perp$ — специальный символ, обозначающий «зацикливание» и не принадлежащий $\mathcal{A}^*$

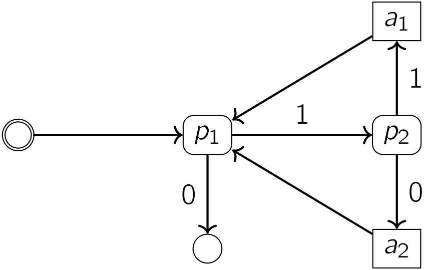
$\mathcal{L}$ — так обозначим множество всех функций разметки (над заданными алфавитами операторных и логических символов)

Семантика схемы описывается отображением $\overline{\pi} : \mathcal{L} \to (\mathcal{A}^* \cup \{\perp\})$, описанным выше

Схемы π_1 и π_2 над одинаковыми множествами операторных и логических символов эквивалентны, если $\overline{\pi}_1 = \overline{\pi}_2$

Проблема эквивалентности схем Ляпунова-Янова формулируется так: для двух произвольно заданных схем Ляпунова-Янова проверить, эквивалентны ли эти схемы

Пример вычисления схемы Ляпунова-Янова



Вычисление этой схемы на функции разметки μ, такой что

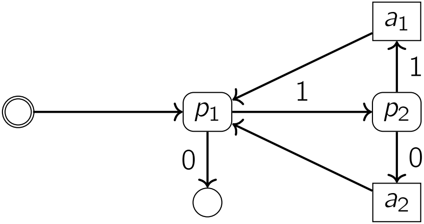
- $\mu(\lambda, p_1) = \mu(\lambda, p_2) = \mu(a_1, p_1) = 1$ и
- $\mu(a_1, p_2) = \mu(a_1 a_2, p_1) = 0$,

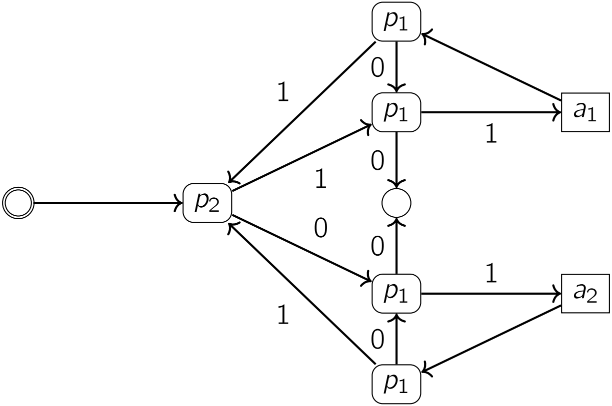
устроено так:

$(\circledcirc, \lambda)$, $(\boxed{p_1}, \lambda)$, $(\boxed{p_2}, \lambda)$, $(\boxed{a_1}, \lambda)$, $(\boxed{p_1}, a_1)$, $(\boxed{p_2}, a_1)$, $(\boxed{a_2}, a_1)$, $(\boxed{p_1}, a_1 a_2)$, $(\bigcirc, a_1 a_2)$

Значит, $\overline{\pi}(\mu) = a_1 a_2$

Пример эквивалентных схем Ляпунова-Янова





Теорема. Проблема эквивалентности схем Ляпунова-Янова разрешима

Доказательство.

Транслируем произвольно заданную схему π в конечный автомат A_π следующим образом

Состояниями автомата являются вход, выход и преобразователи

Начальное состояние — это вход

Заключительное состояние — это выход

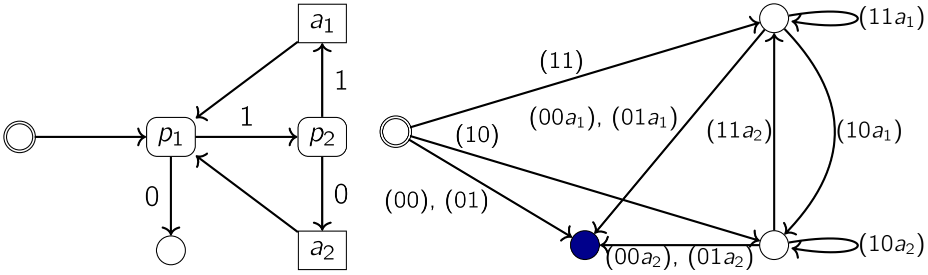
Переход $q_1 \to q_2$ включается в автомат тогда и только тогда, когда существует набор значений $(b_1, \dots, b_n)$ всех логических символов $(p_1, \dots, p_n)$, такой что в π можно попасть из q_1 в q_2, проходя по пути только распознаватели и выбирая для $\boxed{p_i}$ дугу, помеченную b_i

Проверку такой достижимости несложно переформулировать как проверку выполнимости булевой формулы, отвечающей преобразователям, соединяющим q_1 с q_2

Если q_1 — вход, то дуга $q_1 \to q_2$ помечается $(b_1, \dots, b_n)$, а иначе — $q_1 = \boxed{a}$ и дуга $q_1 \to q_2$ помечается $(b_1, \dots, b_n, a)$

Доказательство.

Пример схемы и соответствующего автомата



Несложно убедиться, что схемы Ляпунова-Янова эквивалентны тогда и только тогда, когда эквивалентны соответствующие им автоматы ▼

Схемы Ляпунова-Янова — это очень простая модель с *относительно простым* решением проблемы эквивалентности

Но в этой модели не учитываются многие особенности устройства «реальных» программ и соответствующие взаимосвязи между операторами и логическими условиями

В частности, никак не учитывается наличие переменных и выражений, использующихся в присваиваниях и условиях «реальных» программ

Переход от схем Ляпунова-Янова к схемам, учитывающим эти особенности (стандартным схемам программ), аналогичен переходу от логики высказываний к логике предикатов