Схемы программ (часть2)

Лекция 5

Конфигурация программы

- *Конфигурацией программы* (S, I) называется пара $u=(k, W)$, где k – метка вершины схемы, а W – состояние ее памяти
- Выполнение программы описывается конечной или бесконечной последовательностью конфигураций, которая называется *протоколом выполнения программы*

Формальное определение протокола

- Протокол ($u_0, u_1, \ldots, u_i, u_{i+1}, \ldots$) выполнения программы (S, I) определяется следующим образом:

1. $u_0 = (0, W_0)$, где W_0 - начальное состояние памяти схемы S при интерпретации I. Пусть $u_i = (k_i, W_i)$ - i-я конфигурация, а O – оператор схемы S в вершине k_i.
2. Если O – заключительный оператор **стоп**($т_1, т_2, \ldots, т_n$), то u_i – последняя конфигурация и протокол конечен. Тогда говорят, что программа (S, I) *останавливается*, а последовательность значений $т_{1I}(W_i), т_{2I}(W_i), \ldots, т_{nI}(W_i)$ называется *результатом*

Формальное определение протокола

1. В противном случае в протоколе имеется следующая (i+1)-я конфигурация $u_{i+1}=(k_{i+1}, W_{i+1})$, причем
   - если O – начальный оператор, а выходящая из него дуга ведет к вершине l, то $k_{i+1}=l$, $W_{i+1}=W_i$;
   - если O – оператор присваивания $x:=\tau$, а выходящая из него дуга ведет к вершине l, то $k_{i+1}=l$, $W_{i+1}=^{x} W_i$, $W_{i+1}(x)=\tau_I(W_i)$;
   - если O – условный оператор π и $\pi_I(W_i)=\Delta$, где $\Delta \in \{0,1\}$, а выходящая из этого распознавателя Δ-дуга ведет к вершине l, то $k_{i+1}=l$, $W_{i+1}=W_i$;

Протокол выполнения программы

- Таким образом, программа *останавливается* тогда и только тогда, когда *протокол ее выполнения конечен*
- В противном случае программа *зацикливается* и результат ее выполнения не определен

Схема как алгоритм

- Можно определить интерпретацию как задание только функциональных и предикатных символов
- В этом случае схема описывает алгоритм и определяет частичную функцию из $\mathcal{D}^n$ в $\mathcal{D}^*$, где n – число переменных в схеме (каким-либо образом упорядоченных), а $\mathcal{D}^*$ - множество последовательностей элементов из $\mathcal{D}$
- Такой вариант определения программы больше соответствует общепринятому разделению на собственно программу и исходные данные

Схема как алгоритм

- Однако, для изучения семантических свойств схем программ отделение исходных данных от программы несущественно, т.к. объектом исследования остается схема, а программа является лишь некоторым вспомогательным объектом

Понятия тотальности и пустоты

- Стандартная схема S в базисе $\mathcal{B}$ называется *тотальной*, если для любой интерпретации I базиса $\mathcal{B}$ программа (S, I) останавливается
- Стандартная схема S в базисе $\mathcal{B}$ называется *пустой*, если для любой интерпретации I базиса $\mathcal{B}$ программа (S, I) зацикливается

Отношение эквивалентности для схем

- Отношение эквивалентности вводится для стандартных схем в одном базисе
- Если схемы S_1 и S_2 построены в двух различных базисах $\mathcal{B}_1$ и $\mathcal{B}_2$, то их можно привести к одному базису, в качестве которого взять объединение $\mathcal{B}_1$ и $\mathcal{B}_2$

Отношение эквивалентности для схем

- Говорят, что схемы S_1 и S_2 в базисе $\mathcal{B}$ *функционально эквивалентны* ($S_1 \sim S_2$), если для любой интерпретации I базиса $\mathcal{B}$ программы (S_1, I) и (S_2, I) либо обе зацикливаются, либо обе останавливаются с одинаковым результатом, т.е. $val(S_1, I) \simeq val(S_2, I)$

Цепочки стандартных схем

- *Цепочкой стандартной схемы* называется:
  1. конечный путь по вершинам схемы, идущий от начальной вершины к конечной
  2. бесконечный путь по вершинам, начинающийся от начальной вершины схемы
- В случае, когда вершина v – распознаватель будем снабжать каждое вхождение v в цепочку верхним индексом 0 или 1 в зависимости от того, по какой из исходящих из вершины v дуг продолжается построение цепочки

Цепочки стандартных схем

- Таким образом, цепочку можно записать как последовательность меток вершин, причем некоторые из этих меток имеют верхний индекс 0 или 1:

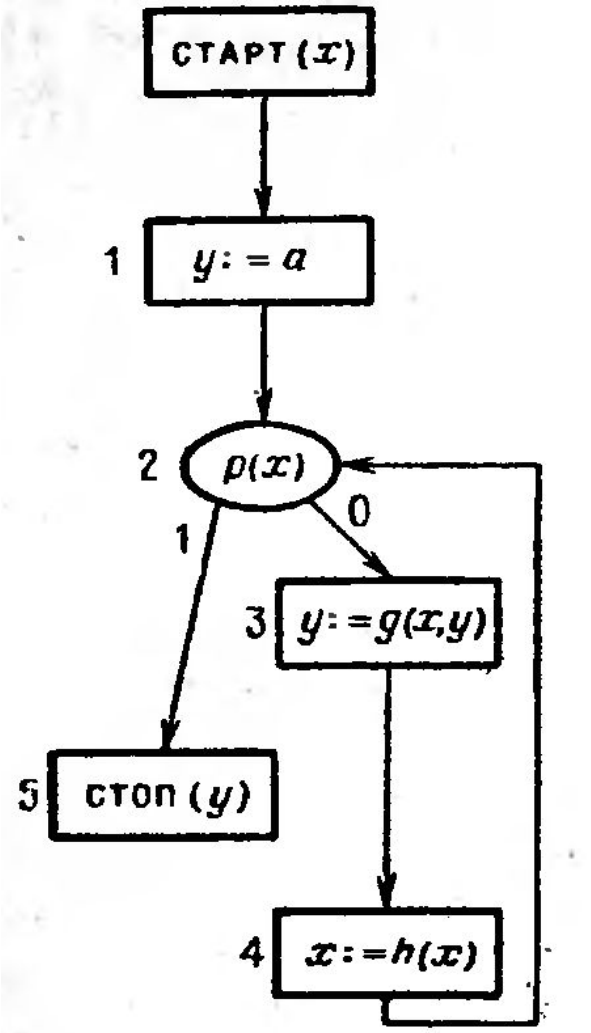
  $(0, 1, 2^1, 5)$

  $(0, 1, 2^0, 3, 4, 2^0, 3, 4, 2, 1, 5)$

  $(0, 1, 2^0, 3, 4, 2^0, \ldots, 2^0, \ldots)$

Цепочки операторов

- *Цепочкой операторов* называется последовательность операторов, метящих вершины некоторой стандартной схемы



Цепочки операторов

- Например, для схемы, представленной на предыдущем слайде, возможны цепочки следующие операторов:

$$(\text{старт}\ (x),\ y := a,\ p^1(x),\ \text{стоп}\ (y))$$
$$(\text{старт}\ (x),\ y := a,\ p^0(x),\ y := g(x, y),\ x := h(x),\ p^0(x),\ y := g(x, y),\ x := h(x),\ p^0(x),\ \ldots,\ p^0(x),\ \ldots).$$

- Предикатные символы в цепочке операторов помечены теми же верхними индексами 0 или 1, которыми помечены соответствующие метки распознавателей в цепочке (в отличие от индексов местности, которые здесь мы будем опускать, индексы 0 и 1 не заключены в скобки)

Допустимые цепочки стандартных схем

- Пусть S – стандартная схема в базисе $\mathcal{B}$, I - некоторая интерпретация базиса $\mathcal{B}$, $(0, 1, k_2, k_3, \ldots)$ - последовательность меток инструкций схемы, выписанных в том порядке, в котором эти метки входят в конфигурации протокола выполнения программы (S, I)
- Эта последовательность является цепочкой схемы S

Допустимые цепочки стандартных схем

- Будем говорить, что интерпретация I *подтверждает* (*порождает*) эту цепочку
- Цепочка стандартной схемы в базисе $\mathcal{B}$ называется *допустимой*, если она порождается хотя бы одной интерпретацией этого базиса

Семантический характер допустимости

- Не всякая цепочка стандартной схемы является допустимой
- Это связано с тем обстоятельством, что понятие цепочки определено *синтаксически*, тогда как свойство допустимости требует привлечения *семантики* в виде определенной *интерпретации схемы*

Свободные стандартные схемы

- Стандартная схема называется *свободной*, если все ее цепочки допустимы
- В тотальной схеме все допустимые цепочки (и соответствующие им цепочки операторов) конечны
- В пустой схеме все допустимые цепочки (и соответствующие им цепочки операторов) бесконечны

Свободные интерпретации

- Отношения тотальности, пустоты и эквивалентности стандартных схем определены с использованием понятия множества всех возможных интерпретаций базиса
- Очевидно, что такие определения не являются конструктивными, т.е. не позволяют на практике установить наличие или отсутствие указанных свойств у той или иной стандартной схемы

Свободные интерпретации

- Однако, существует подкласс интерпретаций, называемый *свободными*, образующий ядро класса всех интерпретаций
- Это означает, что справедливость каких-либо высказываний о семантических свойствах стандартных схем достаточно доказать только для класса программ, получаемых *только с помощью свободных интерпретаций*

Свободные интерпретации

Все свободные интерпретации базиса $\mathcal{B}$ имеют одну и ту же область интерпретации, которая совпадает со множеством T всех термов базиса $\mathcal{B}$. Все свободные интерпретации одинаково интерпретируют переменные и функциональные символы, а именно:

а) для любой переменной x из базиса $\mathcal{B}$ и для любой свободной интерпретации I_h этого базиса $I_h(x) = x$;

б) для любой константы a из базиса $I_h(a) = a$;

в) для любого функционального символа $f^{(n)}$ из базиса $\mathcal{B}$, где $n \geqslant 1$, $I_h(f^{(n)}) = F^{(n)}$: $T^n \to T$, где $F^{(n)}$ — словарная функция такая, что $F^{(n)}(\tau_1, \tau_2, \ldots, \tau_n) = f^{(n)}(\tau_1, \tau_2, \ldots, \tau_n)$, т. е. функция $F^{(n)}$ по термам $\tau_1, \tau_2, \ldots, \tau_n$ из T строит новый терм, используя функциональный символ $f^{(n)}$.

Свободные интерпретации

- Интерпретация предикатных символов, в отличие от интерпретации переменных и функциональных символов, полностью «свободна»: в конкретной свободной интерпретации предикатному символу сопоставляется произвольный предикат, отображающий множество термов T базиса B на множество $\{0,1\}$
- *Итак, разные свободные интерпретации различаются лишь интерпретацией предикатных символов*

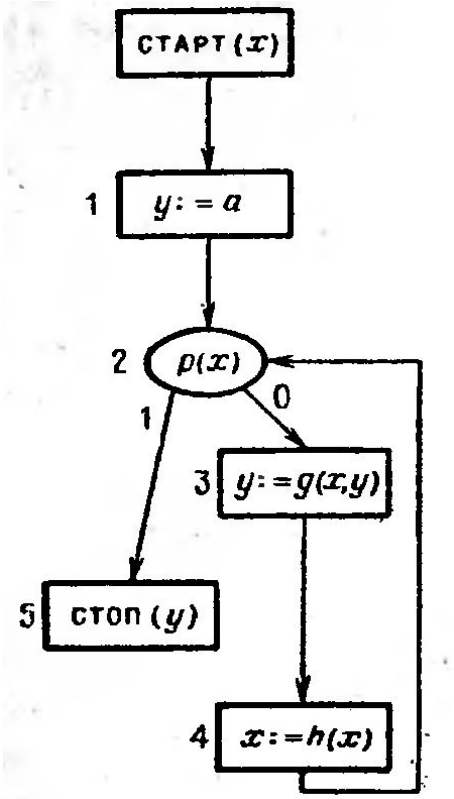
Свободные интерпретации

Таким образом, после введения свободных интерпретаций термы используются в двух разных качествах, как функциональные выражения в схемах и как значения переменных и выражений. Если нам потребуется различать эти два употребления термов, то мы будем термы-значения заключать в апострофы. Так, если $\tau_1 = {}'f(x, a)'$ — терм-значение переменной x, а $\tau_2 = {}'g(y)'$ — терм-значение переменной y, то значение свободно интерпретированного терма-выражения $\tau_3 = f(x, h(y))$ равно терму-значению $'f(f(x, a), h(g(y)))'$.

Пример

- Пусть I_h – свободная интерпретация базиса, в котором определена данная схема
- Определим предикат $p(x)$ следующим образом:

  $p(т) = 1$, если число функциональных символов в т больше 2-х, иначе $p(т) = 0$



Протокол выполнения программы

Конфигурация	Метка	Значения	
		x	y
u_0	0	$'x'$	$'y'$
u_1	1	$'x'$	$'a'$
u_2	2	$'x'$	$'a'$
u_3	3	$'x'$	$'g\,(x,\ a)'$
u_4	4	$'h\,(x)'$	$'g\,(x,\ a)'$
u_5	2	$'h\,(x)'$	$'g\,(x,\ a)'$
u_6	3	$'h\,(x)'$	$'g\,(h\,(x),\ g\,(x,\ a))'$
u_7	4	$'h\,(h\,(x))'$	$'g\,(h\,(x),\ g\,(x,\ a))'$
u_8	2	$'h\,(h\,(x))'$	$'g\,(h\,(x),\ g\,(x,\ a))'$
u_9	3	$'h\,(h\,(x))'$	$'g\,(h\,(h\,(x)),\ g\,(h\,(x),\ g\,(x,a)))'$
u_{10}	4	$'h\,(h\,(h\,(x)))'$	$'g\,(h\,(h\,(x)),\ g\,(h\,(x),\ g\,(x,a)))'$
u_{11}	2	$'h\,(h\,(h\,(x)))'$	$'g\,(h\,(h\,(x)),\ g\,(h\,(x),\ g\,(x,a)))'$
u_{12}	5	$'h\,(h\,(h\,(x)))'$	$'g\,(h\,(h\,(x)),\ g\,(h\,(a),\ g\,(x,a)))'$

Интерпретация термов и тестов

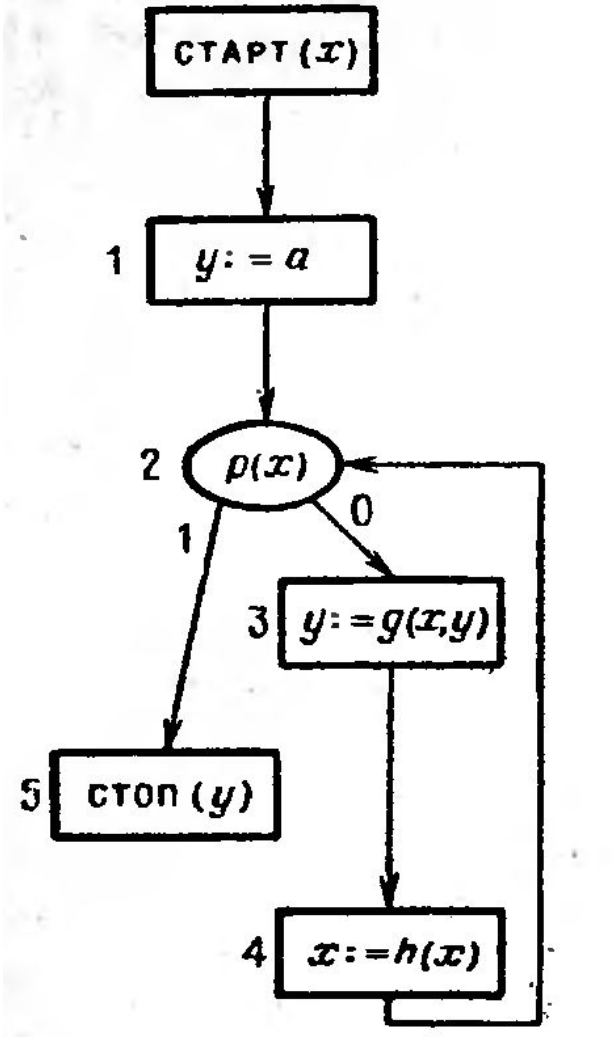
- Интерпретацию термов и тестов определим следующим очевидным образом:

$$I(\tau) = \begin{cases} I(x), \tau = x \\ I(a), \tau = a \\ I\left(f^{(n)}\right)\left(I(\tau_1), \dots, I(\tau_n)\right), \tau = f^{(n)}(\tau_1, \dots \tau_n) \\ I\left(p^{(n)}\right)\left(I(\tau_1), \dots, I(\tau_n)\right), \tau = p^{(n)}(\tau_1, \dots \tau_n) \end{cases}$$

Согласованные свободные интерпретации

- Говорят, что интерпретация I и свободная интерпретация I_h того же базиса $\mathcal{B}$ согласованы, если для любого логического выражения π справедливо $I(\pi) = I_h(\pi)$
- *Лемма: Для каждой интерпретации I базиса $\mathcal{B}$ существует согласованная с ней свободная интерпретация этого базиса*

Пример согласованных интерпретаций



- Интерпретация базиса:

$\mathcal{D}$ – множество целых неотрицательных чисел

$I(x)=3$, $I(y)=1$, $I(a)=1$

$I(g)=\mathcal{G}(d_1,d_2)$, где $\mathcal{G}(d_1,d_2)= d_1 * d_2$

$I(h)=\mathcal{H}(d)$, где $\mathcal{H}(d)=d-1$

$I(p)=\mathcal{P}(d)$, где $\mathcal{P}(d)=1$ при $d=0$ и $\mathcal{P}(d)=0$ при $d>0$

Пример согласованных интерпретаций

- Эта интерпретация согласована с рассмотренной выше свободной интерпретацией данной схемы, поскольку $I(p(\mathsf{T})) = I_h(p(\mathsf{T}))$ для всех возможных термов базиса
- В то же время, изменение интерпретации переменной x с $I(x)=3$ на $I(x)=4$ нарушает указанную согласованность

Теоремы о свободных интерпретациях

Теорема 4.1 (Лакхэм — Парк — Патерсон). *Стандартные схемы S_1 и S_2 в базисе $\mathfrak{B}$ функционально эквивалентны тогда и только тогда, когда они функционально эквивалентны на множестве всех свободных интерпретаций базиса $\mathfrak{B}$, т. е. когда для любой свободной интерпретации I программы (S_1, I) и (S_2, I) либо обе зацикливаются, либо обе останавливаются и* $\mathrm{val}(S_1, I) = \mathrm{val}(S_2, I)$.

Теорема 4.2. *Стандартная схема S в базисе $\mathfrak{B}$ пуста (тотальна) тогда и только тогда, когда она пуста (тотальна) на множестве всех свободных интерпретаций этого базиса, т. е. если для любой свободной интерпретации I_h программа (S, I_h) зацикливается (останавливается).*

Теорема 4.3. *Стандартная схема в базисе $\mathfrak{B}$ свободна тогда и только тогда, когда она свободна на множестве всех свободных интерпретаций этого базиса, т. е. когда каждая цепочка схемы подтверждается хотя бы одной свободной интерпретацией.*

Логико-термальная эквивалентность

Подстановка термов

- Пусть $x_1, x_2, \ldots, x_n \ (n \geq 0)$ – попарно различные переменные, $т_1, т_2, \ldots, т_n$ – термы из множества термов $\mathcal{T}$ базиса схемы
- *Подстановкой термов* в функциональное выражение $f^{(n)}(x_1, x_2, \ldots, x_n)$ называется выражение, получающееся из исходного одновременной заменой каждого из вхождений переменных x_i на терм $т_I$
- Формально такая подстановка будет обозначаться следующим образом:

  $f^{(n)}[т_1/x_1, т_2/x_2, \ldots, т_n/x_n]$
- Аналогичным образом определяется подстановка термов для предикатного выражения p (теста)

Термальное значение переменной

- Определим термальное значение переменной x для конечного пути w схемы S как терм $t(w,x)$, который строится следующим образом:

1. если путь содержит только один оператор A, то $t(w,x)$ = т , если A – оператор присваивания $x := т$, и $t(w,x)$ = x в остальных случаях
2. если $w = w'Ae$, где A – оператор, e – выходящая из него дуга, w' – непустой путь, ведущий к A, а $x_1, x_2, \ldots, x_n$ – все переменные терма $t\,(Ae,x)$, то

$$t(w,x) = t\,(Ae,x)\,[t(w', x_1)/x_1, \ldots, t(w', x_n)/x_n]$$

Термальное значение переменной

- Таким образом, термальное значение переменной x для конечного пути w, завершающегося оператором A, получается из термального значения переменной x для пути Ae заменой переменных, входящих в оператор A, их термальными значениями на отрезке пути w, предшествующем A

Термальное значение терма

- Понятие термального значения очевидным образом распространяется на произвольные термы т:

  если $x_1, x_2, \ldots, x_n$ – все переменные терма т, то положим

$$t(w, \mathrm{т}) = \mathrm{т}[t(w, x_1)/x_1, \ldots, t(w, x_n)/x_n]$$

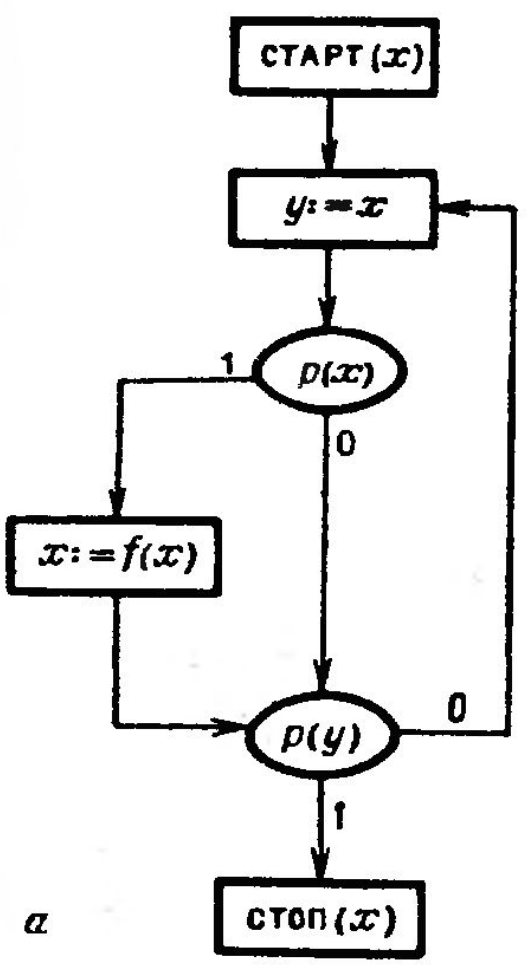
Термальное значение терма

- Например, пути

старт $(x);\ y:=x;\ p^1(x);\ x:=f(x);$

$p^0(y);\ y:=x;\ p^1(x);\ x:=f(x)$

в этой схеме соответствует термальное значение $f(f(x))$ переменной x



Логико-термальная история

Для пути w в некоторой стандартной схеме S определим ее *логико-термальную историю* $\mathrm{lt}\,(S, w)$ как слово, которое строится следующим образом.

1. Если путь w не содержит распознавателей и заключительной вершины, то $\mathrm{lt}\,(S, w)$ — пустое слово.

2. Если $w = w've$, где v — распознаватель с тестом $p\,(\tau_1, \ldots, \tau_k)$, а e — выходящая из него Δ-дуга, $\Delta \in \{0, 1\}$, то

$$\mathrm{lt}\,(S, w) = \mathrm{lt}\,(S, w')\, p^{\Delta}\,(t\,(w', \tau_1), \ldots, t\,(w', \tau_k)).$$

3. Если $w = w'v$, где v — заключительная вершина с оператором стоп $(\tau_1, \ldots, \tau_k)$, то

$$\mathrm{lt}\,(S, w) = \mathrm{lt}\,(S, w')\, t\,(w', \tau_1) \ldots t\,(w', \tau_k).$$

Например, логико-термальной историей пути, упомянутого в приведенном выше примере, будет

$$p^1\,(x)\, p^0\,(x)\, p^1\,(f\,(x)).$$

Детерминант стандартной схемы

- *Детерминантом* (обозначение: $det(S)$) стандартной схемы S называется множество логико-термальных историй всех цепочек этой схемы, завершающихся заключительным оператором
- Говорят, что интерпретация I стандартной схемы S согласована с логико-термальной историей $lt(S,w)$ для некоторого пути этой схемы, если цепочка операторов, соответствующая пути w, подтверждается этой интерпретацией

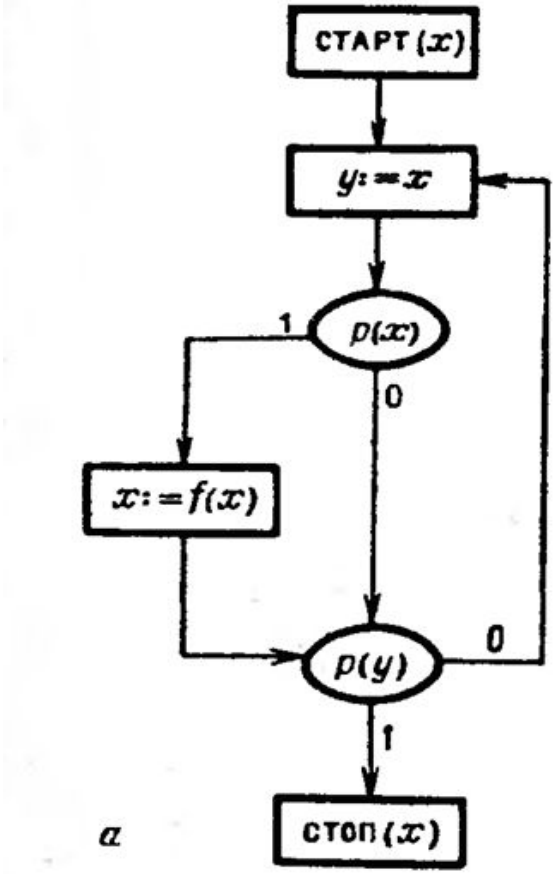
Логико-термальная эквивалентность стандартных схем

- Очевидно, что любая интерпретация может быть согласована не более чем с одной логико-термальной историей из $det(S)$
- Схемы S_1 и S_2 называются *логико-термально эквивалентными* (сокращенно *лт-эквивалентными*; обозначение: $S_1 \overset{лт}{\sim} S_2$), если их детерминанты совпадают
- Логико-термально эквивалентные схемы являются функционально эквивалентными

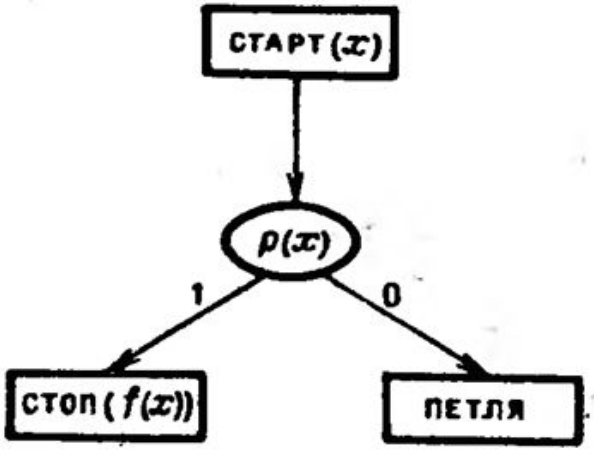
$$S_1 \overset{лт}{\sim} S_2 \Rightarrow S_1 \sim S_2$$

Логико-термальная и функциональная эквивалентность стандартных схем

- Обратное утверждение неверно, что подтверждается примером



а



б

Логико-термальная и функциональная эквивалентность стандартных схем

- Действительно, при p(x) = 0 любая свободная интерпретация для схемы a приводит к возникновению петли, показанной на схеме б
- При p(x) = 1 любая свободная интерпретация для схемы a приводит к завершению выполнения соответствующей программы со значением f(x), совпадающим со значением на схеме б
- В то же время, детерминант стандартной схемы a содержит логико-термальные истории для бесконечного числа путей, тогда как детерминант схемы б состоит из единственной лт-истории