Задача проверки модели (Model checking)

Евтушенко Н.В. **Винарский Е.М.**

*по всем вопросам писать на **vinevg2015@gmail.com** или в телеграмм @evgenii1996*

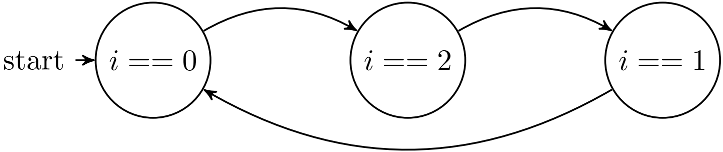
7 ноября 2024 г.

“Текущий” пример

Algorithm 1 ”Текущий пример”

$i = 0$
for *true* **do**
    $i = i + 2$
    $i = i - 1$
    $i = i - 1$
end for

Нас интересует следующее свойство:
Переменная i бесконечно много раз принимает значение 2



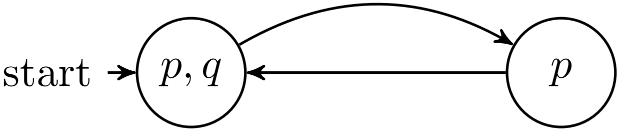
события: $\{i = 0;\ i = 2;\ i = 1\}$

vinevg2015@gmail.com

Структура Крипке

Пусть AP – множество *атомарных высказываний*, тогда под структурой Крипке будем понимать систему $M = (S, S_0, \rightarrow, L)$

- S – множество состояний
- $S_0 \subseteq S$ – множество начальных состояний
- $\rightarrow \subseteq S \times S$ – *тотальное*[1] отношение переходов
- $L : S \rightarrow 2^{AP}$ – функция разметки состояний

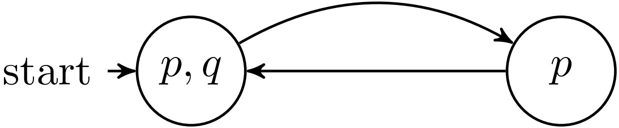


$$L(s_0) = \{p, q\}, L(s_1) = \{p\}$$

[1] отношение переходов *тотально*, если для любого состояния $s \in S$ существует $s' \in S$ такое, что существует переход из s в s'

vinevg2015@gmail.com

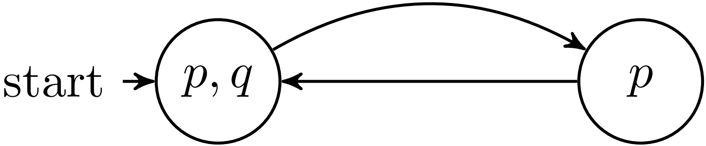
Трассы, порождаемые структурой Крипке



- Путь π из состояния s – это бесконечная последовательность состояний вида: $s \to s_1 \to s_2 \ldots$
- Трасса $\alpha(\pi)$ пути π – это бесконечная последовательность событий: $L(s)L(s_1)L(s_2)\ldots$
- $\Pi(M)$ – множество всех путей из начальных состояний структуры Крипке M
- $Tr(M) = \{\alpha(\pi) \mid \pi \in \Pi(M)\}$

vinevg2015@gmail.com

Трассы, порождаемые структурой Крипке (Пример)



- $L(s_0) = \{p, q\}$
- $L(s_1) = \{p\}$
- Путь π из состояния s_0 : $s_0 \to s_1 \to s_0 \to \dots$
- Трасса $\alpha(\pi)$ пути π: $\{p, q\}\{p\}\{p, q\} \dots$

vinevg2015@gmail.com

Логика линейного времени (LTL)

Пусть AP – множество *атомарных высказываний*, тогда LTL-формула φ строится по следующим правилам:

- $\varphi = a$, где $a \in AP$
- φ – LTL-формула, тогда $\neg\varphi$ – LTL-формула
- φ_1 и φ_2 – LTL-формулы, тогда $\varphi_1 \wedge \varphi_2$ – LTL-формула
- φ – LTL-формула, тогда $\mathbf{X}\varphi$ – LTL-формула
- φ – LTL-формула, тогда $\mathbf{F}\varphi$ – LTL-формула
- φ – LTL-формула, тогда $\mathbf{G}\varphi$ – LTL-формула

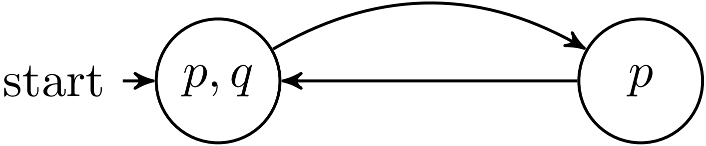
Логика линейного времени (LTL)

- AP – множество *атомарных высказываний*
- Трасса τ – (возможно) бесконечная последовательность событий
- $\tau[i]$ – i-ое событие трассы τ
- τ^i – суффикс трассы τ, начинающийся с i-ого события

Тогда *отношение выполнимости* формулы φ на трассе τ определяется следующим образом:

- $\tau \models a \Leftrightarrow a \in \tau[0]$
- $\tau \models \psi_1 \wedge \psi_2 \Leftrightarrow \tau \models \psi_1 \wedge \tau \models \psi_2$
- $\tau \models \mathbf{X}\varphi \Leftrightarrow \tau^1 \models \varphi$
- $\tau \models \psi_1 \mathbf{U} \psi_2 \Leftrightarrow \exists k, k \geq 0$: $\tau^k \models \psi_2$, $\tau^m \models \psi_1$ для всех $m \in [0, k)$
- $\tau \models \mathbf{F}\phi \Leftrightarrow \exists k \geq 0$: $\tau^k \models \varphi$
- $\tau \models \mathbf{G}\varphi \Leftrightarrow \forall k \geq 0$: $\tau^k \models \varphi$

vinevg2015@gmail.com



$\varphi = \mathbf{GF}q$. Верно ли, что в любой трассе структуры Крипке $\mathcal{M}$ событие q будет встречаться бесконечно-часто?
$\tau = (\{p, q\}, \{p\})^\omega$. Верно ли, что $\tau \models \mathbf{GF}q$? То есть верно ли, что $\forall k \geq 0, \tau^k \models \mathbf{F}q$?
В структуре Крипке $\mathcal{M}$

- $\tau^k = \{p, q\}, \{p\}, \{p, q\}, \{p\}, \ldots$ если $k = 2 * \ell$
- $\tau^k = \{p\}, \{p, q\}, \{p\}, \{p, q\}, \ldots$ если $k = 2 * \ell + 1$

Верно ли, что $\forall k \geq 0 \; \exists m > k : q \in \tau[m]$?

- Если мы отправили запрос, то когда-нибудь в будущем обязательно получим ответ:

vinevg2015@gmail.com

Логика линейного времени (LTL)

Примеры

- Если мы отправили запрос, то когда-нибудь в будущем обязательно получим ответ:
  $\mathbf{G}(request \Rightarrow \mathbf{F}reply)$
- Если мы отправили сообщение, то не сможем отправить следующее, до тех пор, пока не получим ответ:

- Если мы отправили запрос, то когда-нибудь в будущем обязательно получим ответ:
  $\mathbf{G}(\mathit{request} \Rightarrow \mathbf{F}\mathit{reply})$
- Если мы отправили сообщение, то не сможем отправить следующее, до тех пор, пока не получим ответ:
  $\mathbf{G}(\mathit{send} \Rightarrow \mathbf{X}(\neg \mathit{send}\mathbf{U}\mathit{receive}))$
- Флаг, отвечающий за то, что система никогда не будет находиться в “тупиковой” ситуации никогда не поднят:

- Если мы отправили запрос, то когда-нибудь в будущем обязательно получим ответ:
  $\mathbf{G}(request \Rightarrow \mathbf{F}reply)$
- Если мы отправили сообщение, то не сможем отправить следующее, до тех пор, пока не получим ответ:
  $\mathbf{G}(send \Rightarrow \mathbf{X}(\neg send \mathbf{U} receive))$
- Флаг, отвечающий за то, что система никогда не будет находиться в "тупиковой" ситуации никогда не поднят:
  $\mathbf{G}(deadlock\_flag == false)$
- Если система послала сообщение, то ответ будет обязательно получен, и не наступит момента, когда мы больше не сможем отправлять сообщения:

vinevg2015@gmail.com

- Если мы отправили запрос, то когда-нибудь в будущем обязательно получим ответ:
  $\mathbf{G}(request \Rightarrow \mathbf{F}reply)$
- Если мы отправили сообщение, то не сможем отправить следующее, до тех пор, пока не получим ответ:
  $\mathbf{G}(send \Rightarrow \mathbf{X}(\neg send\,\mathbf{U}\,receive))$
- Флаг, отвечающий за то, что система никогда не будет находиться в “тупиковой” ситуации никогда не поднят:
  $\mathbf{G}(deadlock\_flag == false)$
- Если система послала сообщение, то ответ будет обязательно получен, и не наступит момента, когда мы больше не сможем отправлять сообщения:
  $\mathbf{G}((send \Rightarrow \mathbf{XF}receive) \wedge \mathbf{F}send)$

vinevg2015@gmail.com

Свойства вычисления программ

- Каждое вычисление системы характеризуется трассой, то есть последовательностью событий
- Поведение системы характеризуется свойством, то есть множеством трасс

Формально:

- AP – конечное множество атомарных предикатов
- *Событие* E – любое множество атомарных предикатов, $E \subseteq AP$
- *Трасса* α – любая бесконечная последовательность событий, $\alpha \in (2^{AP})^{\omega}$
- *Вычислительное свойство* P – любое трасс, то есть $P \subseteq (2^{AP})^{\omega}$

vinevg2015@gmail.com

Свойства вычисления программ (2)

- $M = (S, S_0, \to, L)$ – структура Крипке
- $\Pi(M)$ – множество всех путей из начальных состояний структуры Крипке M
- $Tr(M) = \{\alpha(\pi) \mid \pi \in \Pi(M)\}$ – множество всех трасс, порождаемых структурой Крипке M

Структура Крипке M *удовлетворяет свойству* P ($M \models P$), если

$$Tr(M) \subseteq P$$

vinevg2015@gmail.com

Свойство безопасности (safety)

Неформально:

- "ничего плохого не произойдёт"
- "Если случится что-то плохое, это уже никак не исправить"

Свойство P_{safe} – *свойство безопасности*, если для каждой трассы $\sigma, \sigma \in (2^{AP})^{\omega} \backslash P_{safe}$ такой, что у σ существует конечный префикс β такой, что $\beta.\sigma' \notin P_{safe}$ для любой трассы σ'

Примеры:

- Несколько процессов не могут одновременно войти в одну критическую секцию
- Сообщение не может быть потеряно при передачи

vinevg2015@gmail.com

Свойство живости (liveness)

Неформально:

- "что-нибудь хорошее обязательно произойдёт"
- "Цель будет достигнута, независимо от того, что было ранее"

Свойство P_{live} – *свойство живости*, если для каждой конечной трассы $\beta, \beta \in (2^{AP})^*$ существует $\sigma \in (2^{AP})^\omega$ такая, что $\beta.\sigma \in (2^{AP})^\omega$

Примеры:

- Процессов может входить в критическую секцию бесконечное количество раз
- Сообщение когда-нибудь будет доставлено

vinevg2015@gmail.com

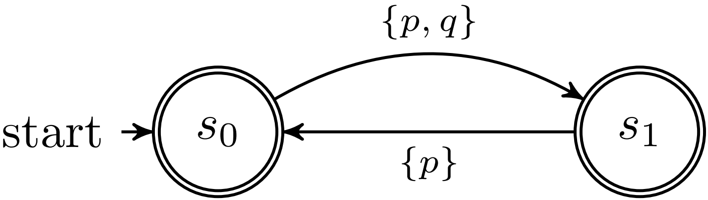
Автоматы Бюхи (1)

Пусть $\Sigma = \{\sigma_1, \ldots, \sigma_m\}$ – конечный алфавит, тогда под автомтом Бюхи будем понимать систему $M = (S, S_0, \to, L)$

- S – множество состояний
- $S_0 \subseteq S$ – конечное непустое множество начальных состояний
- $\to \subseteq S \times \Sigma \times S$ – отношение переходов
- $F \subseteq S$ – конечное непустое множество финальных состояний

Автоматы Бюхи (2)

- Автомат Бюхи работает с бесконечными словами вида $\sigma_1\sigma_2\ldots\sigma_n\ldots$, где $\sigma_i \in \Sigma$
- *Трасса автомата Бюхи* – это бесконечная последовательность вида: $run = s_0 \xrightarrow{\sigma_1} s_1 \xrightarrow{\sigma_2} s_2 \xrightarrow{\sigma_3} s_3$
- слово принимается автоматом Бюхи, если и только если $inf(run)^2 \cap F \neq \emptyset$



[2] $inf(run)$ – состояния, встречающиеся бесконечно часто на трассе run

Задача Model Checking

- Пусть имеем *LTL*-формулу φ и структуру Крипке M
- *LTL*-формула φ выполняется на пути π в M ($M, \pi \models \phi$), если $\alpha(\pi) \models \phi$
- *LTL*-формула φ выполняется на модели M ($M \models \varphi$), если она выполняется на каждом пути π множества $Tr(M)$, т.е. $Tr(M) \subseteq Tr(\varphi)$

Задача *Model Checking* – проверить справедливость соотношения $M \models \varphi$

vinevg2015@gmail.com

Схема решения задачи Model Checking

1. По модели M строится автомат Бюхи A_M, распознающий множество бесконечных трасс $Tr(M)$
2. Строится отрицание формулы φ, затем по ней строится автомат $A_{\neg\varphi}$, распознающий множество бесконечных трасс $Tr(\neg\varphi)$
3. Строится автомат A, распознающий множество бесконечных трасс $Tr(M) \cap Tr(\neg\varphi)$
4. Анализируется язык, распознаваемый автоматом Бюхи A $Tr(M) \cap Tr(\neg\varphi)$
   - если язык, распознаваемый A – **пустой**, то $M \models \varphi$
   - если язык, распознаваемый A – **НЕ пустой**, то $M \not\models \varphi$ и слова, принадлежащие этому языку – контрпримеры

vinevg2015@gmail.com