Задача проверки модели

Евтушенко Н.В., Винарский Е.М.

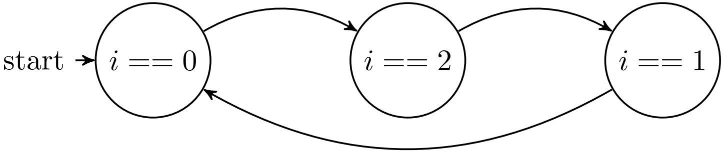
по всем вопросам писать на vinevg2015@gmail.com

8 ноября 2018 г.

Содержание

Algorithm 1 "Бегущий пример"

```
i = 0
for true do
    i = i + 2
    i = i - 1
    i = i - 1
end for
```



LTL-формулы

- *Трасса run* – бесконечная последовательность событий
- *run*[j] – *j*-ое событие трассы τ
- *run*^{j} – суффикс трассы τ, начинающийся с *j*-ого события
- *AP* – множество *атомарных высказываний*

В "Бегущем примере"

- события – $\{i = 0; i = 1; i = 2\}$
- $AP = \{[i == 0], [i == 1], [i == 2]\}$

LTL-формула

- $\phi = a \in AP$
- ϕ – *LTL*-формула, тогда $\neg\phi$ – *LTL*-формула
- ϕ_1, ϕ_2 – *LTL*-формулы, тогда $\phi_1 \wedge \phi_2$ – *LTL*-формула
- ϕ – *LTL*-формула, тогда $\mathbb{X}\phi$ – *LTL*-формула
- ϕ_1, ϕ_2 – *LTL*-формулы, тогда $\phi_1 \mathbb{U} \phi_2$ – *LTL*-формула
- ϕ – *LTL*-формула, тогда $\mathbb{F}\phi$ – *LTL*-формула
- ϕ – *LTL*-формула, тогда $\mathbb{G}\phi$ – *LTL*-формула

LTL-формулы (2)

- $run \models a$: выполняется если и только если $a \in run[0]$
- $run \models \neg\phi$: выполняется если и только если $run \not\models \phi$
- $run \models \psi_1 \wedge \psi_2$: выполняется если и только если $run \models \psi_1$ и $run \models \psi_1$
- $run \models \mathbb{X}\phi$: выполняется если и только если $run^1 \models \phi$ (формула выполнима в следующий момент времени)
- $run \models \psi_1 \mathbb{U} \psi_2$: выполняется если и только если $\exists k, k \geq 0$: $run^k \models \psi_2$ $run^m \models \psi_1$ для всех $m \in [0, k)$ (в какой-то момент времени выполнится ϕ_2, а до этого всегда выполняется ϕ_1)
- $run \models \mathbb{F}\phi$: выполняется если и только если $\exists k \geq 0$: $run^k \models \phi$ (когда-то в будущем выполнится ϕ)
- $run \models \mathbb{G}\phi$: выполняется если и только если $\forall k \geq 0$: $run^k \models \phi$ (всегда в будущем выполнится ϕ)

LTL-формулы (примеры)

LTL-формулы описывают требования к верифицируемой системе

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

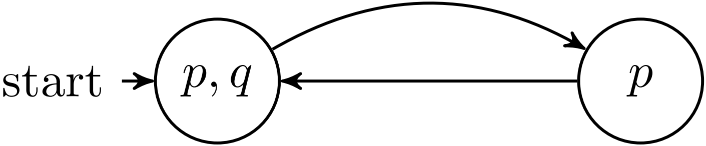
Структура Крипке

- Требования к системе описываются на языке *LTL*-формул
- Для формального доказательства факта, что программа удовлетворяет этим требованиям, необходима формальная модель программы
- В качестве такой модели выступает структура Крипке
- Задача проверки модели заключается в проверки выполнимости формулы на структуре Крипке

AP – множество атомарных высказываний
Структура Крипке – система $M = (S, S_0, \to, L)$

- S – конечное непустое множество состояний
- $S_0 \subseteq S$ – конечное непустое множество начальных состояний
- $\to \subseteq S \times S$ – *тотальное* отношение переходов
- $L : S \to 2^{AP}$ – функция разметки состояний

Отношение переходов *тотальное*, если для любого состояния $s \in S$ существует $s' \in S$ такое, что существует переход из s в s'



Проверка модели (Model Checking)

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

Задача проверки модели

ϕ – LTL-формула, M – структура Крипке

- Формула ϕ *выполняется на пути* π в M ($M, \pi \models \phi$), если $\alpha(\pi) \models \phi$
- Формула ϕ *выполняется на структуре* M ($M \models \phi$), если она выполняется на каждом пути π множества $Tr(M)$, т.е. $Tr(M) \subseteq Tr(\phi)$

Задача Model Checking – проверить справедливость соотношения $M \models \phi$

Схема решения задачи проверки модели

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 $A_{\neg\phi}$, распознающий множество бесконечных трасс $Tr(\neg\phi)$
3. Строится автомат A, распознающий множество бесконечных трасс $Tr(M) \cap Tr(\neg\phi)$
4. анализируется язык, распознаваемый автоматом Бюхи $Tr(M) \cap Tr(\neg\phi)$
   - если язык, распознаваемый A – пустой, то $M \models \phi$
   - если язык, распознаваемый A – НЕ пустой, то $M \not\models \phi$ и слова, принадлежащие этому языку – контрпримеры

Автоматы Бюхи

Автоматом Бюхи (над алфавитом Σ) называется система $A = (S, S_0, \rightarrow, F)$

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

Автоматы Бюхи

inf(run) – состояния, встречающиеся бесконечно часто на трассе *run*

- Автомат Бюхи работает с бесконечными словами вида $\sigma_1\sigma_2\ldots\sigma_n\ldots$, где $\sigma_i \in \Sigma$
- Трасса автомата Бюхи – бесконечная последовательность состояний вида: $run = s_0 \xrightarrow{\sigma_1} s_1 \xrightarrow{\sigma_1} \ldots s_{n-1} \xrightarrow{\sigma_n} s_n \ldots$
- слово принимается автоматом Бюхи если и только если $inf(run) \cap F \neq \emptyset$