

# Задача проверки модели (Model checking)

Евтушенко Н.В.    Винарский Е.М.

*по всем вопросам писать на **[vinevg2015@gmail.com](mailto:vinevg2015@gmail.com)** или в телеграмм **[@evgenii1996](https://t.me/evgenii1996)***

11 ноября 2022 г.

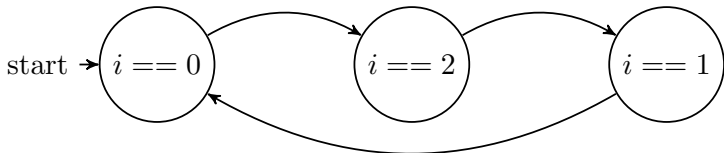
# “Текущий” пример

## Algorithm 1 “Текущий пример”

```
1:  $i = 0$   
2: for true do  
3:    $i = i + 2$   
4:    $i = i - 1$   
5:    $i = i - 1$   
6: end for
```

Нас интересует следующее свойство:

*Переменная  $i$  бесконечно много раз принимает значение 2*

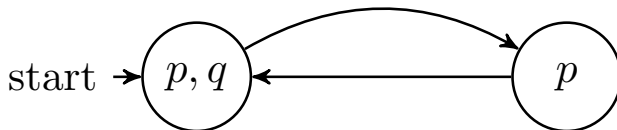


события:  $\{i = 0; i = 2; i = 1\}$

# Структура Крипке

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

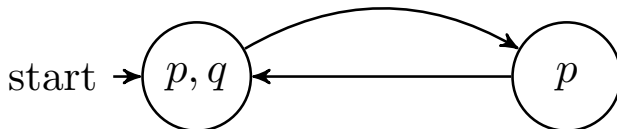
- $S$  – множество состояний
- $S_0 \subseteq S$  – множество начальных состояний
- $\rightarrow \subseteq S \times S$  – *тотальное*<sup>1</sup> отношение переходов
- $L : S \rightarrow 2^{AP}$  – функция разметки состояний



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

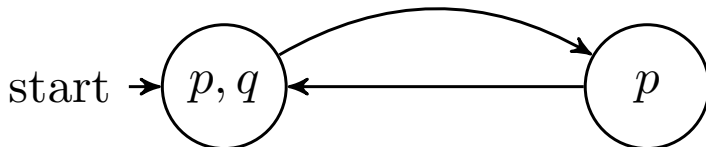
<sup>1</sup>отношение переходов *тотально*, если для любого состояния  $s \in S$  существует  $s' \in S$  такое, что существует переход из  $s$  в  $s'$

# Трассы, порождаемые структурой Крипке



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

# Трассы, порождаемые структурой Крипке (Пример)



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

# Логика линейного времени (LTL)

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

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

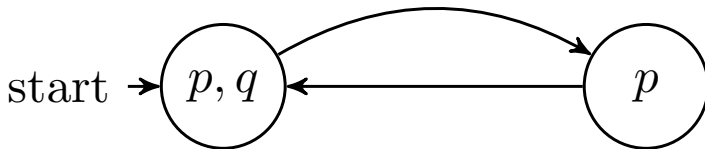
# Логика линейного времени (LTL)

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

Тогда *отношение выполнимости* формулы  $\varphi$  на трассе  $\tau$  определяется следующим образом:

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

# Логика линейного времени (Пример)



$\varphi = \mathbf{GF}q$ . Верно ли, что в любой трассе структуры Крипке  $\mathcal{M}$  событие  $q$  будет встречаться бесконечно-часто?

$\tau = (\{p, q\}, \{p\})^\omega$ . Верно ли, что  $\tau = \mathbf{GF}q$ ? То есть верно ли, что  $\forall k \geq 0, \tau^k \models \mathbf{F}q$ ?

В структуре Крипке  $\mathcal{M}$

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

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



# Логика линейного времени (LTL)

## Примеры

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

# 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:

# 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:  
 $\mathbf{G}(send \Rightarrow \mathbf{X}(\neg send \mathbf{U} receive))$
- Флаг, отвечающий за то, что система никогда не будет находиться в “тупиковой” ситуации никогда не поднят:

# 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:  
 $\mathbf{G}(send \Rightarrow \mathbf{X}(\neg send \mathbf{U} receive))$
- 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:

# Логика линейного времени (LTL)

## Примеры

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

- 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**  $\alpha$  – любая бесконечная последовательность событий,  $\alpha \in (2^{AP})^\omega$
- **Вычислительное свойство**  $P$  – любое трасс, то есть  $P \subseteq (2^{AP})^\omega$

## Свойства вычисления программ (2)

- $M = (S, S_0, \rightarrow, L)$  – 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  
$$Tr(M) \subseteq P$$

# 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  $\sigma, \sigma \in (2^{AP})^\omega \setminus P_{safe}$  такой, что у  $\sigma$  существует конечный префикс  $\beta$  такой, что  $\beta.\sigma' \notin P_{safe}$  для любой трассы  $\sigma'$

Примеры:

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



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

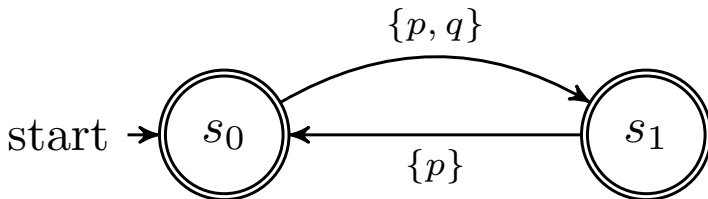
# Автоматы Бюхи (1)

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

- $S$  – 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 (2)

- Автомат Бюхи работает с бесконечными словами вида  $\sigma_1\sigma_2\dots\sigma_n\dots$ , 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  $inf(run)^2 \cap F \neq \emptyset$



<sup>2</sup> $inf(run)$  – состояния, встречающиеся бесконечно часто на трассе  $run$

# Задача Model Checking

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

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

# Схема решения задачи Model Checking

- ❶ По модели  $M$  строится автомат Бюхи  $A_M$ , распознающий множество бесконечных трасс  $Tr(M)$
- ❷ Строится отрицание формулы  $\varphi$ , затем по ней строится автомат  $A_{\neg\varphi}$ , распознающий множество бесконечных трасс  $Tr(\neg\varphi)$
- ❸ Строится автомат  $A$ , распознающий множество бесконечных трасс  $Tr(M) \cap Tr(\neg\varphi)$
- ❹ 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