Математические методы верификации схем и программ

Лекторы:

Захаров Владимир Анатольевич

Подымов Владислав Васильевич

e-mail рассказчика:

valdus@yandex.ru

Осень 2016

Лекция 7

Задача model checking для LTL

Алгоритмы верификации LTL-формул: табличный алгоритм, автоматный алгоритм

Автоматы Бюхи, их свойства и обобщения

Задача model checking для LTL

Напоминание

AP — множество атомарных высказываний

Модель Крипке над множеством AP — это система $M = (S, S_0, R, L)$, где

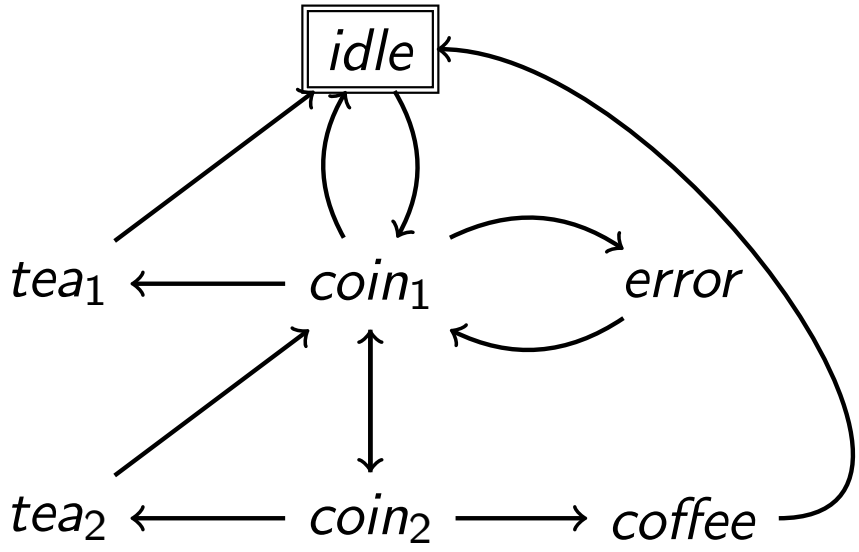
- S — конечное множество состояний
- $S_0 \subseteq S$ — множество начальных состояний
- $R \subseteq S \times S$ — тотальное отношение переходов
- $L : S \to 2^{AP}$ — функция разметки

Отношение переходов тотально, если для любого состояния $s \in S$ существует состояние $s' \in S$, такое что $R(s, s')$

$s \to s'$ — синоним $R(s, s')$

Задача model checking для LTL

Пример: кофейный автомат



$L(idle) = \{idle\}$
$L(coin_1) = \{OneCoin, ReadyForTea\}$
$L(error) = \{OneCoin, ShowErrorMessage\}$
$L(tea_1) = \{GiveTea\}$
$L(coin_2) = \{TwoCoins, ReadyForTea, ReadyForCoffee\}$
$L(coffee) = \{GiveCoffee\}$
$L(tea_2) = \{GiveTea, OneCoin\}$

Задача model checking для LTL

Напоминание

LTL-формула ($ltlf$) имеет следующий вид:

$$ltlf ::= a \mid ltlf \vee ltlf \mid ltlf \,\&\, ltlf \mid \neg ltlf \mid ltlf \rightarrow ltlf \mid \mathbf{X}ltlf \mid \mathbf{F}ltlf \mid \mathbf{G}ltlf \mid ltlf\,\mathbf{U}\,ltlf \mid ltlf\,\mathbf{R}\,ltlf \qquad (a \in AP)$$

В лекции 4 давалось немного другое определение LTL-формул

Отличие — отсутствует символ **A**, с которого *в лекции 4* начиналась каждая LTL-формула

Если ограничиться рассмотрением только логики линейного времени, то этот символ избыточен и опускается

Задача model checking для LTL

Примеры LTL-спецификаций:

- $\mathbf{F}\,\mathit{GiveCoffee}$
- $\mathbf{G}(\mathit{TwoCoins} \rightarrow \neg\mathbf{X}\,\mathit{ShowErrorMessage})$
- $\mathbf{G}\neg(\mathit{OneCoin}\ \&\ \mathbf{X}\,\mathit{GiveCoffee})$
- $\mathbf{GF}(\mathit{GiveTea} \vee \mathit{GiveCoffee})$
- $\mathbf{G}\neg\mathit{ShowErrorMessage} \rightarrow \mathbf{GF}(\mathit{GiveTea} \vee \mathit{GiveCoffee})$
- $\mathbf{G}(\mathit{OneCoin} \vee \mathit{TwoCoins} \vee \mathit{GiveTea} \vee \mathit{GiveCoffee}\,\mathbf{U}\,\mathit{idle})$

А какие из этих требований выполнены для предложенного кофейного автомата?

Задача model checking для LTL

Формулировка задачи model checking для LTL: для заданных модели Крипке M и LTL-формулы φ проверить соотношение

$$M \models \varphi$$

А как это расшифровывается?

Проверить, что любая (*бесконечная*) трасса системы M удовлетворяет свойству φ

Если имеется LTL-формула φ, то для любой трассы любой модели Крипке можно проверить, удовлетворяет ли трасса свойству φ

Иными словами, каждой LTL-формулой φ опредеяется множество трасс $Tr(\varphi)$, таких что какая бы ни была модель M, верно

$$M \models \varphi \Leftrightarrow Tr(M) \subseteq Tr(\varphi)$$

Сравнение выразительных возможностей LTL и CTL

А имеет ли смысл рассматривать логику линейного времени, если имеются "хорошие" алгоритмы проверки CTL-формул?

Утверждение. **Существует LTL-формула, которой не эквивалентна ни одна CTL-формула** (*лекция 4*)

А может быть, мы зря отдельно рассматривали логику ветвящегося времени?

Утверждение. **Существует CTL-формула, которой не эквивалентна ни одна LTL-формула** (*лекция 4*)

(формулы φ, ψ логики CTL* эквивалентны, если для любой модели Крипке M верно либо $M \models \varphi$ и $M \models \psi$, либо $M \not\models \varphi$ и $M \not\models \psi$)

Табличный алгоритм model checking для LTL

Можно ли описать алгоритм разметки состояний подформулами, похожий на табличный алгоритм для CTL?

При попытке это сделать возникает такая проблема:

- табличный алгоритм размечает состояния формулами, **выполняющимися в этих состояниях**
- **AF**φ: при построении всех трасс будет **достигнуто состояние**, в котором верна формула φ
- **F**φ: в любой трассе наступит **момент времени**, когда будет выполнена формула φ
- момент времени, подразумеваемый формулой **F**φ, не связан напрямую ни с каким состоянием

Несоответствие моментов времени и состояний усложняет принципы работы табличного алгоритма для LTL

Однако эти трудности могут быть преодолены

Небольшая пауза

1. Все ли помнят, что табличный алгоритм model checking для LTL уже рассказывался?
2. Все ли помнят, как выглядело описание алгоритма?
3. Все ли представляют, как этот алгоритм работает хоть для каких-нибудь LTL-формулы и модели Крипке?
4. Все ли помнят, **почему** этот алгоритм работает?

Всё это рассказывалось давно, и тогда не было такого понимания задачи model checking, как есть сейчас

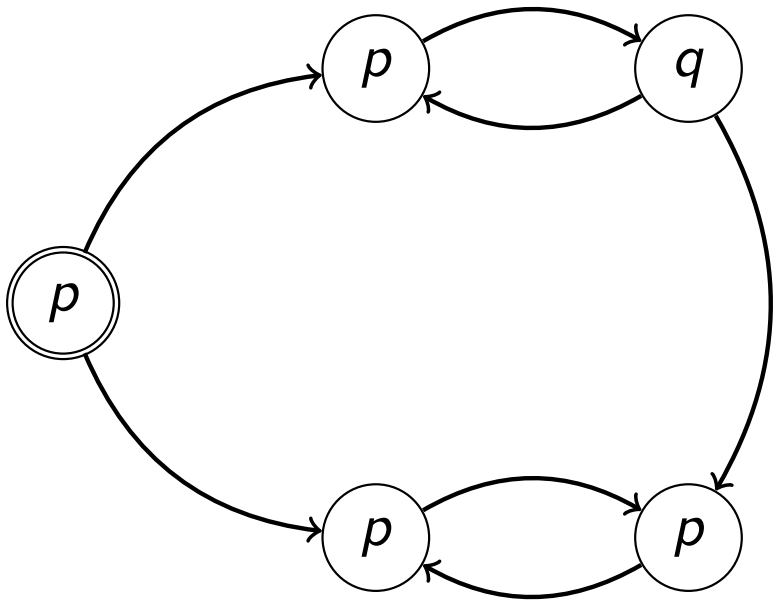
Поэтому можно повторить, как выглядит табличный алгоритм проверки LTL-формул

Табличный алгоритм model checking для LTL

Пример:

φ: $p\mathbf{U}q$

M:



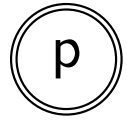
Верно ли, что $M \models \varphi$?

Табличный алгоритм model checking для LTL

Попытаемся доказать, что $M \not\models \varphi$

Будем обходить состояния модели и *компактно описывать всевозможные способы работы системы*, размечая состояния множествами формул, истинных в данный момент работы

Отталкиваясь от предположения о том, какие формулы истинны и ложны для текущего состояния в текущий момент времени (***сегодня***), будем строить предположения о следующем моменте времени (***завтра***), переходя к следующему состоянию

Например, находясь в состоянии 

, мы точно знаем: p, q

Кроме того, можно **предположить** как $p\mathbf{U}q$, так и $p\mathbf{U}q$

Табличный алгоритм model checking для LTL

Чтобы согласовать предположения об истинности формул сегодня и завтра, будем также делать предположения вида $\mathbf{X}\varphi$ для некоторых формул φ

Зачем так усложнять себе жизнь?

Например, справедлива такая эквивалентность:

$$p\mathbf{U}q = q \vee p \,\&\, \mathbf{X}(p\mathbf{U}q)$$

Если предположить сегодня $\mathbf{X}(p\mathbf{U}q)$, то

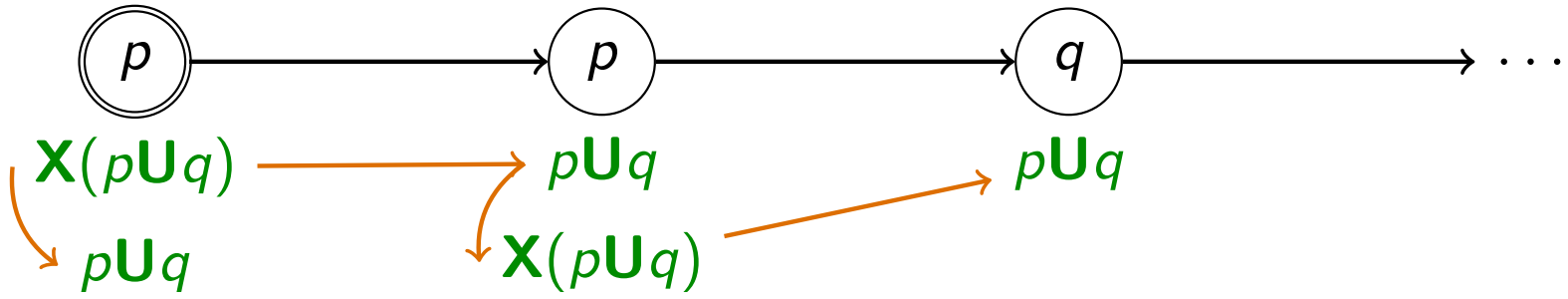
- сегодня $p\mathbf{U}q$ $\Leftrightarrow$ сегодня p или q
- завтра мы обязаны предположить $p\mathbf{U}q$

Если предположить сегодня $\mathbf{X}(p\mathbf{U}q)$, то

- сегодня $p\mathbf{U}q$ $\Leftrightarrow$ сегодня q
- завтра мы обязаны предположить $p\mathbf{U}q$

Табличный алгоритм model checking для LTL

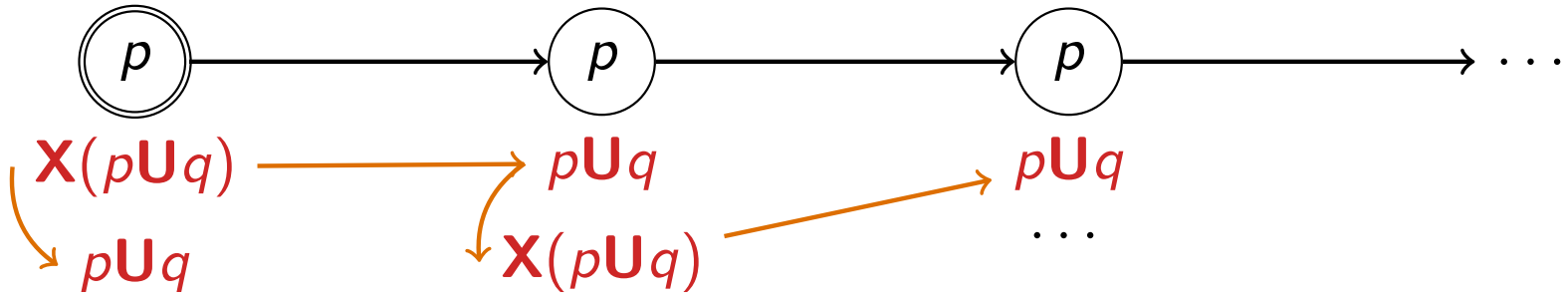
Пример:



Успех!: доказано, что для любой трассы, начинающейся с этих трёх состояний, верно $p\mathbf{U}q$

Табличный алгоритм model checking для LTL

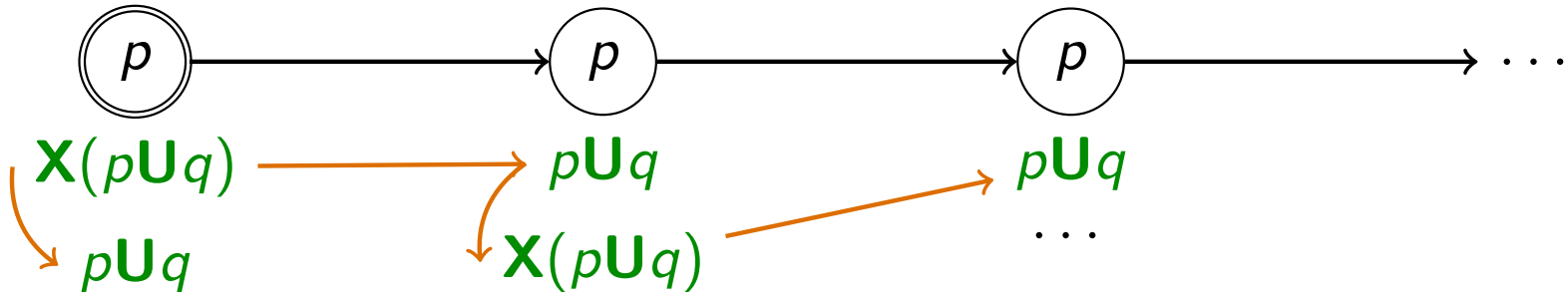
Пример:



Успех!: доказано, что для любой трассы, в которой всегда верно p и неверно q, неверно и $p\mathbf{U}q$

Табличный алгоритм model checking для LTL

Пример:



Неуспех?

- Согласно предположениям, верно $p\mathbf{U}q$
- Для этой трассы свойство $p\mathbf{U}q$ не выполнено

Как можно описать особенность строящихся предположений, из-за которой они не совпали с реальными свойствами трассы?

Почти всегда предполагается q и $\mathbf{X}(p\mathbf{U}q)$

Табличный алгоритм model checking для LTL

Чтобы исключить неуспешное построение предположений, достаточно следить за тем, что именно мы предполагаем для посылок, следствий и обещаний на завтра для формул вида $\varphi \mathbf{U} \psi$

Для формулы $\varphi \mathbf{U} \psi$ сегодня звенит звонок, если для текущего предположения верно хотя бы одно из двух: ψ или $\mathbf{X}(\varphi \mathbf{U} \psi)$

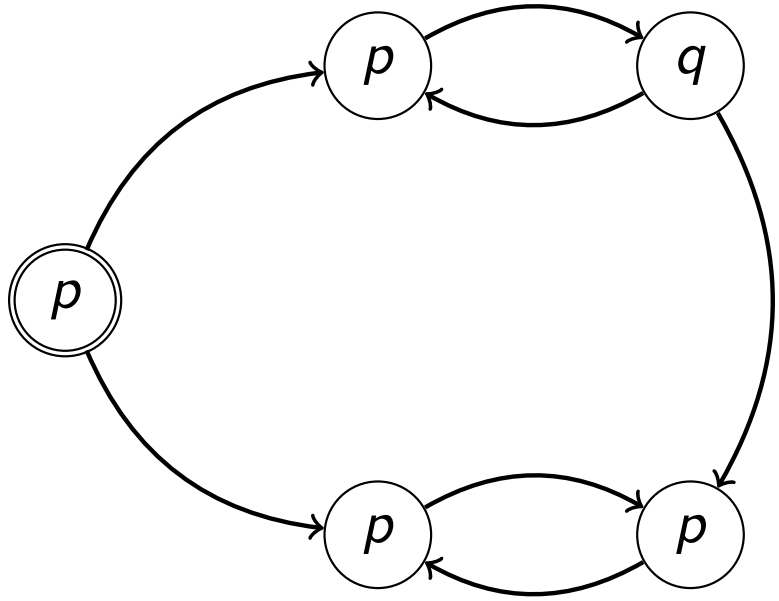
В допустимой последовательности предположений для каждой подформулы вида $\varphi \mathbf{U} \psi$ или $\varphi \mathbf{R} \psi$ проверяемой формулы звонок должен звенеть бесконечно часто

Табличный алгоритм model checking для LTL

Вернёмся к примеру:

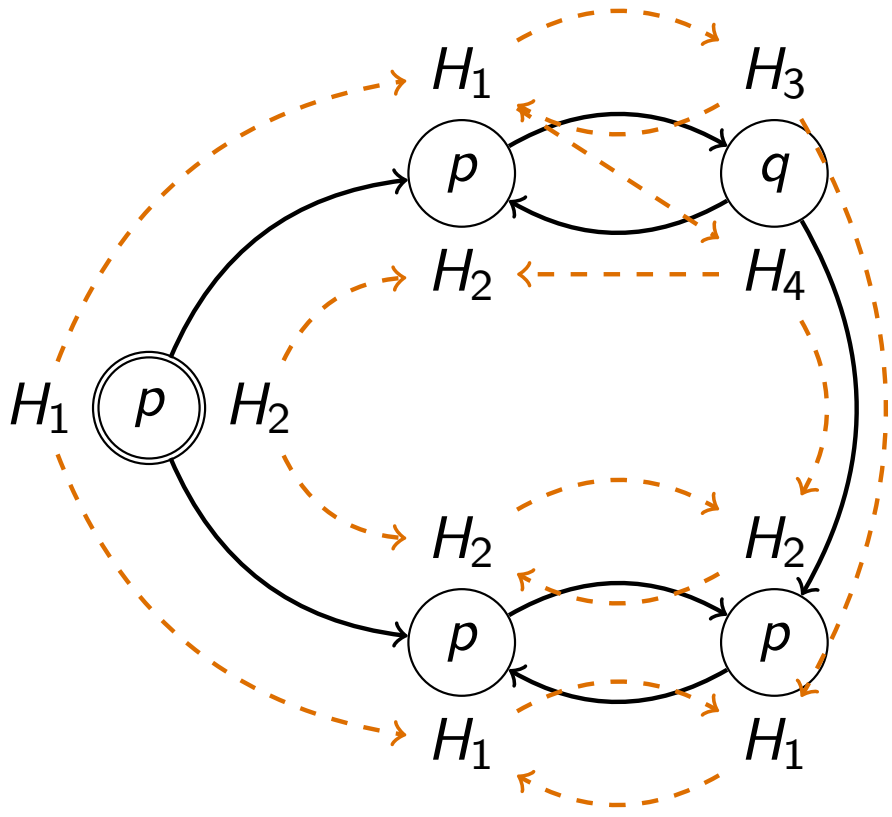
φ: $p\mathbf{U}q$

M:



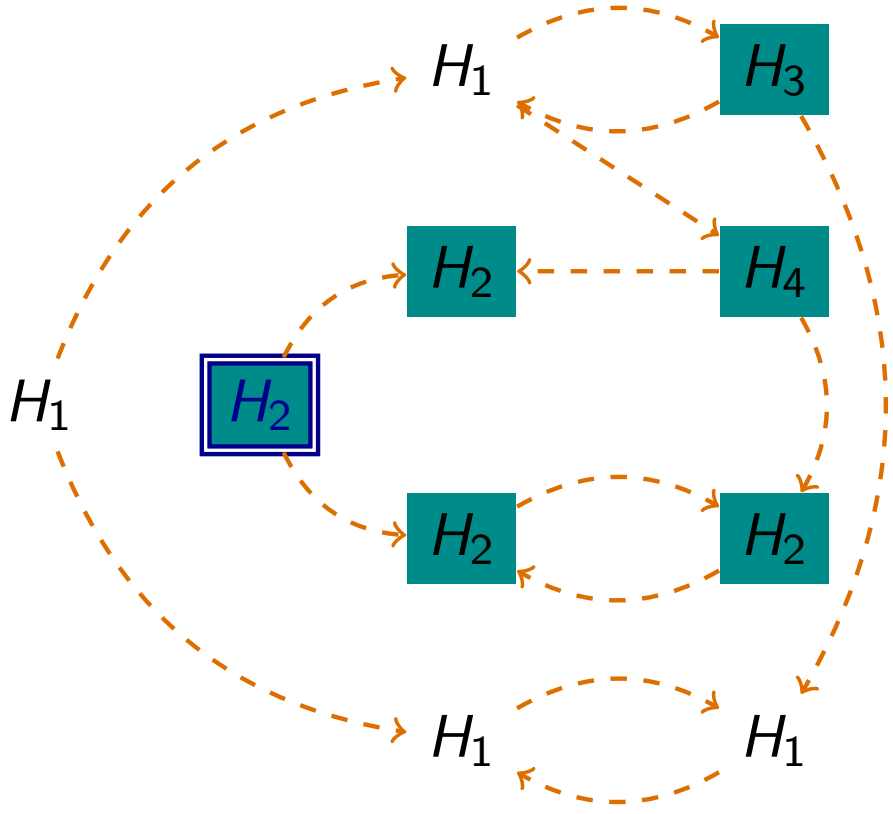
Попробуем разметить состояния модели предположениями всевозможными способами

Табличный алгоритм model checking для LTL



$H_1 = \{p, q, \mathbf{X}(p\mathbf{U}q), p\mathbf{U}q\}$
$H_2 = \{p, q, \mathbf{X}(p\mathbf{U}q), p\mathbf{U}q\}$
$H_3 = \{p, q, \mathbf{X}(p\mathbf{U}q), p\mathbf{U}q\}$
$H_4 = \{p, q, \mathbf{X}(p\mathbf{U}q), p\mathbf{U}q\}$

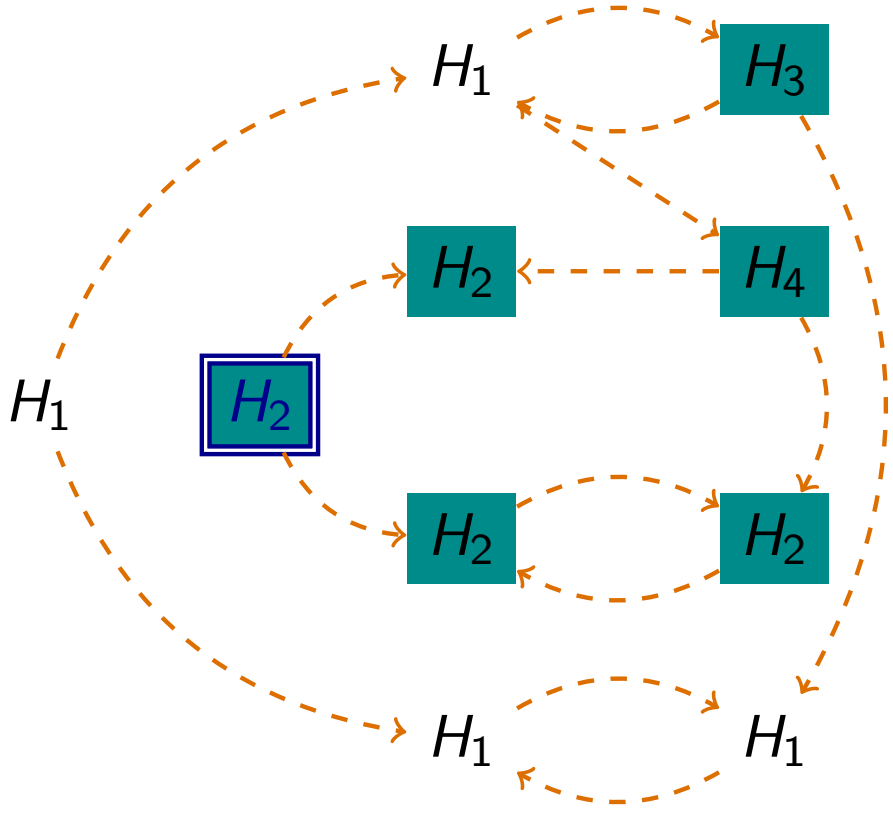
Табличный алгоритм model checking для LTL



◎ — предположение соответствует начальному состоянию и содержит $p\mathbf{U}q$

● — звенит звонок для $p\mathbf{U}q$

Табличный алгоритм model checking для LTL

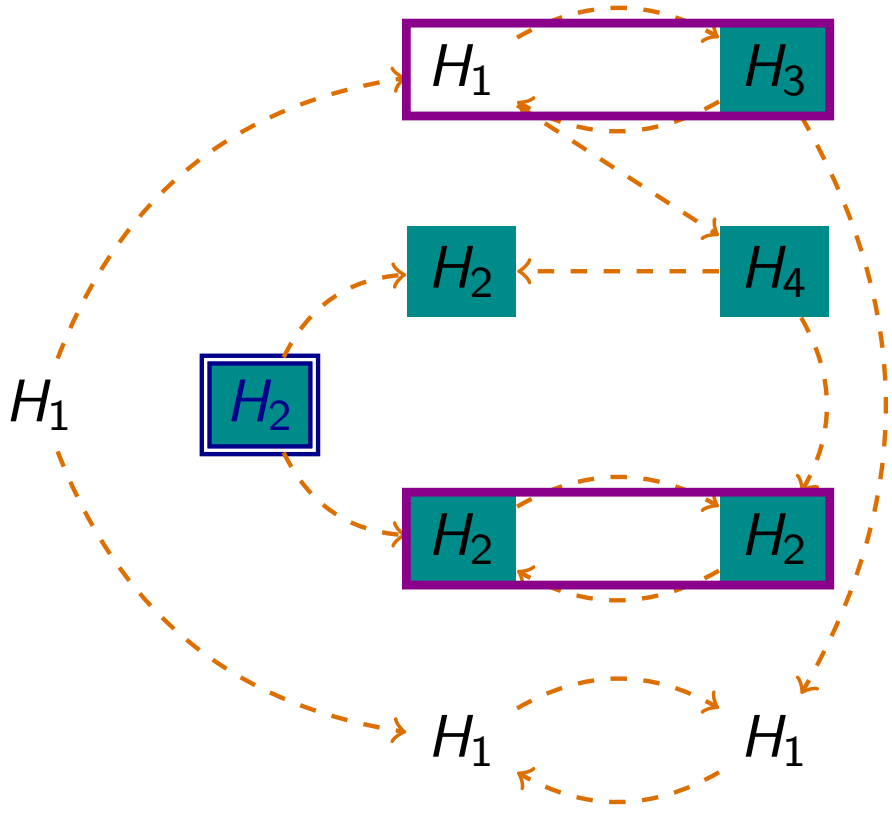


$M \not\models \varphi$

$\Leftrightarrow$

из ◎ исходит маршрут, в котором звонок звенит бесконечно часто

Табличный алгоритм model checking для LTL

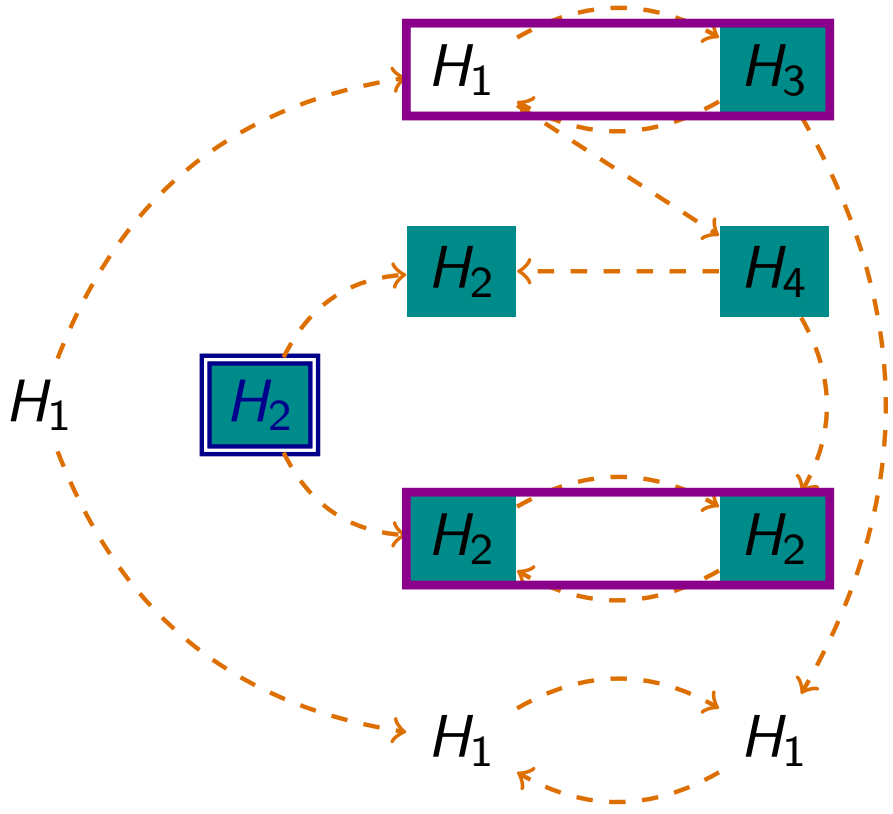


$M \not\models \varphi$

$\Leftrightarrow$

из ◎ достижима компонента сильной связности, содержащая хотя бы одну вершину, в которой звенит звонок

Табличный алгоритм model checking для LTL



Итог: $M \not\models p\mathbf{U}q$

Табличный алгоритм model checking для LTL

Более полное описание табличного алгоритма выглядит так:

- исключим из формулы φ операторы $\rightarrow, \vee, \mathbf{F}, \mathbf{G}$, оставив только $\&, \neg, \mathbf{X}, \mathbf{U}$, и уберём все двойные отрицания
- построим множество всех формул, которые могут появиться при разметке состояний модели Крипке предположениями
- для каждого состояния s определим множество "хороших" для этого состояния предположений $\mathcal{H}_s$, и заменим состояние s на совокупность состояний (s, H), $H \in \mathcal{H}_s$
- проведём всевозможными способами дуги $(s_1, H_1) \rightarrow (s_2, H_2)$ так, чтобы выполнялось $s_1 \rightarrow s_2$ и при этом предположения на сегодня (H_1) и завтра (H_2) не противоречили друг другу

Табличный алгоритм model checking для LTL

Более полное описание табличного алгоритма выглядит так:

- в расклеенной таким образом модели найдём компоненты сильной связности, такие что для каждой формулы вида $\psi \mathbf{U} \chi$, которая может появиться в предположениях, хотя бы в одном состоянии звенит звонок
- проверим, найдётся ли состояние (s, H), такое что
  - s — начальное состояние исходной модели,
  - проверяемая формула предполагается ложной в H
  - из состояния (s, H) достижима хотя бы одна из найденных компонент сильной связности

Табличный алгоритм model checking для LTL

Преобразование формулы

Чтобы привести формулу к нужному виду, достаточно использовать такие эквивалентные преобразования:

- $\mathbf{G}\varphi = \neg\mathbf{F}\neg\varphi$
- $\mathbf{F}\varphi = \mathbf{true}\mathbf{U}\varphi$
- $\varphi\mathbf{R}\psi = \neg(\neg\varphi\mathbf{U}\neg\psi)$
- $\varphi \to \psi = \neg\varphi \vee \psi$
- $\varphi \vee \psi = \neg(\neg\varphi \,\&\, \neg\psi)$
- $\neg\neg\varphi = \varphi$

Табличный алгоритм model checking для LTL

Замыкание Фишера-Ладнера $[\varphi]_{FL}$ формулы φ — это множество всех формул, которые используются при разметке состояний модели Крипке предположениями для проверки формулы φ, и его можно описать так:

- каждая подформула формулы φ, не имеющая отрицание внешней связкой, входит в $[\varphi]_{FL}$
- если $(\psi \mathbf{U} \chi) \in [\varphi]_{FL}$, то $\mathbf{X}(\psi \mathbf{U} \chi) \in [\varphi]_{FL}$

Предположение — это множество $[\varphi]_{FL}$, в котором каждая подформула покрашена в зелёный или красный цвет (считается истинной или ложной соответственно)

Для краткости будем использовать запись $\neg\psi$ как синоним ψ, а запись $\neg\psi$ — как синоним ψ

Табличный алгоритм model checking для LTL

Хорошее предположение H для заданного состояния s модели Крипке (S, S_0, R, L) удовлетворяет двум условиям:

1. для любого атомарного высказывания a верно: $a \in H \Leftrightarrow a \in L(s)$ (согласованность с атомарными высказываниями $L(s)$, или согласованность с состоянием s)
2. внутренняя согласованность: для любых формул $\psi_1 \,\&\, \psi_2, \chi_1 \mathbf{U} \chi_2 \in [\varphi]_{FL}$ верно:
   - $\psi_1 \,\&\, \psi_2 \in H \Leftrightarrow \psi_1, \psi_2 \in H$
   - $\chi_1 \mathbf{U} \chi_2 \in H \Leftrightarrow \chi_2 \in H$ или $\{\chi_1, \mathbf{X}(\chi_1 \mathbf{U} \chi_2)\} \subseteq H$

Не противоречащие друг другу предположения H_1, H_2 на сегодня и завтра внешне согласованы: для любой формулы $\mathbf{X}\psi \in [\varphi]_{FL}$ верно:

$$\mathbf{X}\psi \in H_1 \Leftrightarrow \psi \in H_2$$

При разметке состояний модели Крипке предположениями

- выбираются всевозможные предположения, согласованные внутренне и с размечаемым состоянием
- дуги проводятся для всевозможных пар внешне согласованных предположений

Табличный алгоритм model checking для LTL

Табличный алгоритм model checking для LTL можно считать полностью описанным

Обоснование корректности этого алгоритма непросто для понимания и приведено **в лекциях по курсу “Математическая логика и логическое программирование”**

Достоинства табличного алгоритма: . . .

Недостатки табличного алгоритма:

- труден для понимания
- требуется явный обход графа, полученного из модели Крипке разметкой состояний предположениями (а можно ли этого избежать?)
- состояния могут размечаться большими множествами формул (а насколько большими?)

Алгоритмы model checking

Задача model checking для CTL имеет альтернативу табличному алгоритму: *символьный алгоритм*

Задача model checking для LTL также имеет альтернативу табличному алгоритму: автоматный алгоритм

Если программное средство производит верификацию для CTL, то *скорее всего* в основе реализованного алгоритма верификации лежит символьный алгоритм

Если программное средство производит верификацию для LTL, то *скорее всего* в основе реализованного алгоритма верификации лежит автоматный алгоритм

Автоматный алгоритм model checking для LTL

Общая схема работы автоматного алгоритма:

- имеются модель Крипке M и LTL-формула φ
- построить автомат A_M, описывающий в точности множество всех трасс модели M
- построить автомат $A_{\neg\varphi}$, описывающий в точности множество всех трасс, **не** входящих в свойство φ
- построить автомат $A_M \oplus A_{\neg\varphi}$, описывающий в точности множество всех трасс модели M, не обладающих свойством φ
- проверить, описывается ли автоматом $A_M \oplus A_{\neg\varphi}$ пустое множество трасс
  - если да, то $M \models \varphi$
  - если нет, то $M \not\models \varphi$

И что же это за автоматы?

Автоматы Бюхи

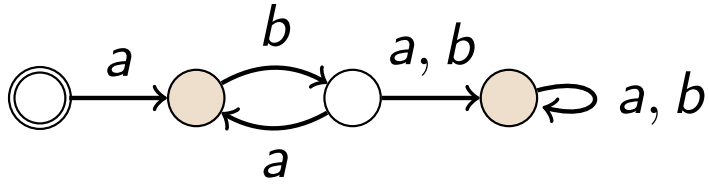
Автомат Бюхи над алфавитом Σ — это система $A = (S, S_0, \delta, F)$, где

- S — конечное множество состояний
- $S_0 \subseteq S$ — множество начальных состояний
- $\delta \subseteq S \times \Sigma \times S$ — отношение переходов
- $F \subseteq S$ — множество заключительных состояний

$s \xrightarrow{\sigma_1,\ldots,\sigma_k} s'$ — синоним записи $\delta(s, \sigma_1, s') \,\&\, \ldots \,\&\, \delta(s, \sigma_k, s')$

Автоматы Бюхи

Пример



◎ — начальное состояние

● — заключительное состояние

Ничего не напоминает?

Синтаксис автоматов Бюхи в точности совпадает с синтаксисом конечных недетерминированных автоматов-распознавателей

Автомат Бюхи, прочитывая конечные слова, изменяет свои состояния точно так же, как и автоматы распознаватели

Отличие этих двух моделей — в семантике: распознаваемом **языке** (множестве слов)

Автоматы Бюхи

Автоматы Бюхи строятся для того, чтобы проверять, содержит ли модель Крипке **трассы**, не удовлетворяющие заданному **свойству**

И трассы, и свойства основаны на **бесконечных** словах

Язык, распознаваемый автоматом Бюхи, — это множество **бесконечных** слов в заданном алфавите

Множество всех бесконечных слов в алфавите Σ будем обозначать так: Σ^ω

Трасса автомата Бюхи $A = (S, S_0, \delta, F)$, порождаемая бесконечным словом $\sigma_1\sigma_2\sigma_3\ldots$ — это любая бесконечная последовательность его состояний вида

$$s_0 \xrightarrow{\sigma_1} s_1 \xrightarrow{\sigma_2} s_2 \xrightarrow{\sigma_3} \ldots,$$

такая что $s_0 \in S_0$

$Tr(A, w)$ — множество всех трасс автомата Бюхи A, порождаемых бесконечным словом w

Автоматы Бюхи

inf(t) — множество состояний, встречающихся бесконечно часто в трассе t автомата Бюхи

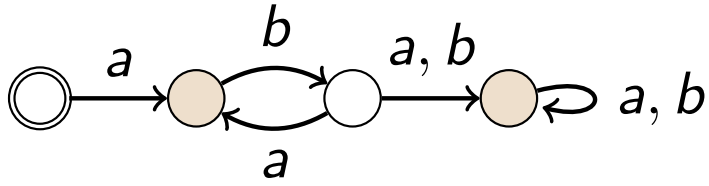
Язык $L(A)$, распознаваемый автоматом Бюхи $A = (S, S_0, \delta, F)$, определяется так:

$$L(A) = \{w \mid \exists t \in Tr(A, w) : inf(t) \cap F \neq \emptyset\}$$

Проще говоря, бесконечное слово w распознаётся автоматом Бюхи, если среди всех трасс автомата, получаемых при прочитывании этого слова, есть такая, в которой бесконечно часто повторяется хотя бы одно заключительное состояние

Автоматы Бюхи

Пример



Какой язык распознаётся этим автоматом Бюхи?

Этот язык состоит в точности из всех бесконечных слов следующего вида:

- $(ab)^\omega$ — слово, получающееся бесконечным повторением слова ab
- $(ab)^+(a|b)^\omega$ — слово, состоящее из хотя бы одного повторения слова ab, за которым следует любое бесконечное сочетание букв a, b

Автоматы Бюхи

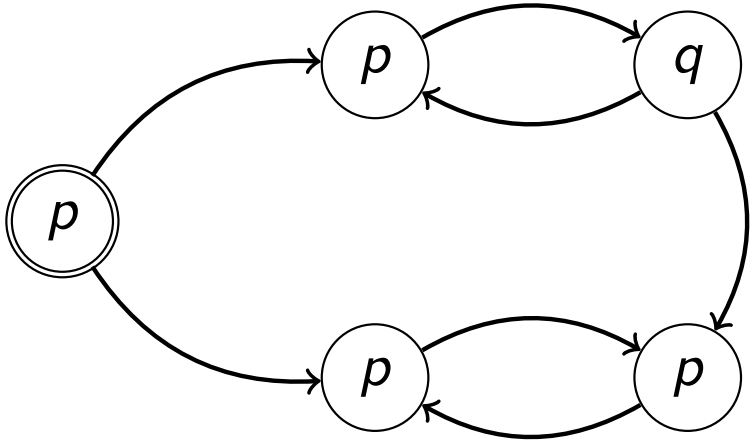
А как построить автомат Бюхи, распознающий в точности множество всех трасс заданной модели Крипке?

Очень просто:

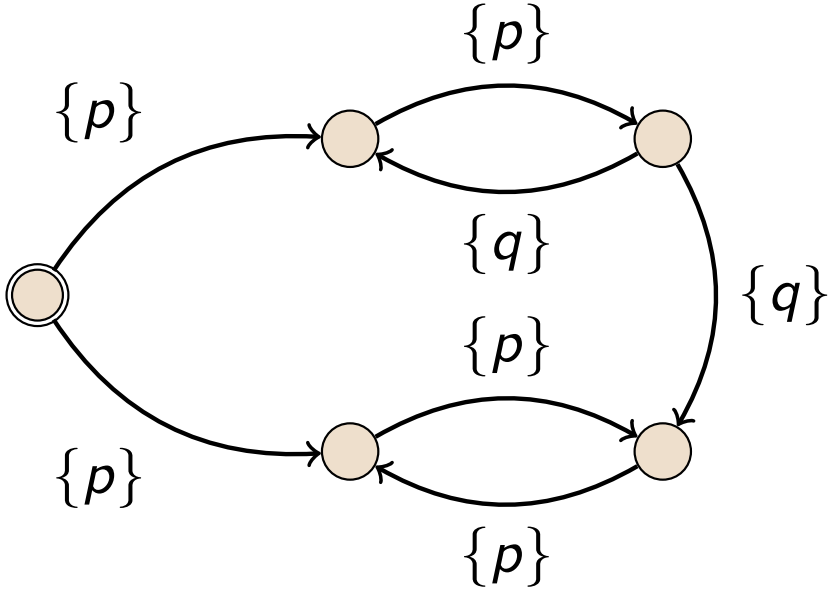
- алфавит, над которым строится автомат: 2^{AP}
- объявим состояния модели состояниями автомата Бюхи
- начальные состояния модели объявим начальными состояниями автомата
- соединим состояния автомата дугами так же, как они были соединены в модели
- множество высказываний, помечавшее состояние модели, переместим на каждую исходящую из состояния дугу в автомате
- объявим **все** состояния автомата заключительными

Автоматы Бюхи

Пример



Автомат, распознающий в точности множество всех трасс этой модели Крипке, может выглядеть так:

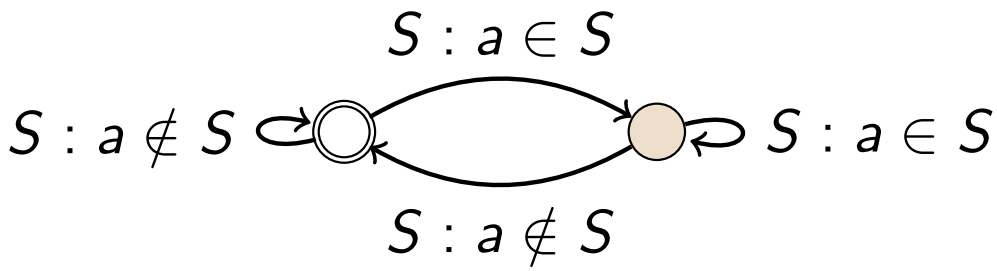


Автоматы Бюхи

А как построить автомат Бюхи, распознающий в точности множество $Tr(\varphi)$ для LTL-формулы φ?

Начнём с примеров

GFa

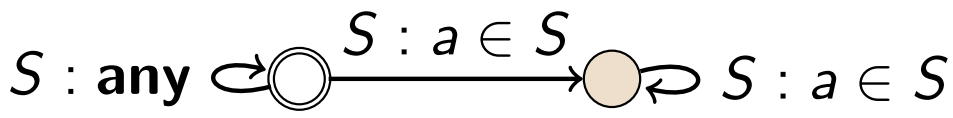


Автоматы Бюхи

А как построить автомат Бюхи, распознающий в точности множество $Tr(\varphi)$ для LTL-формулы φ?

Начнём с примеров

FGa

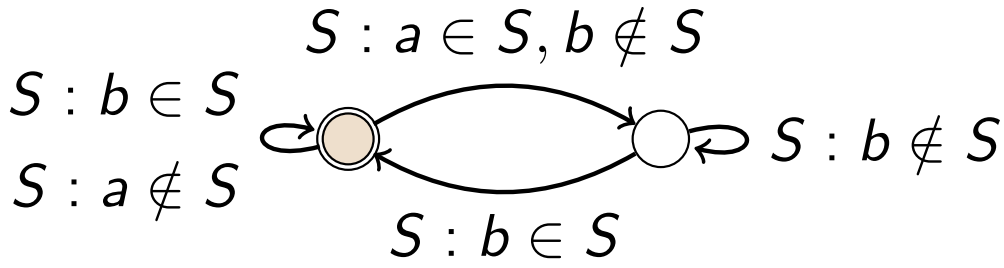


Автоматы Бюхи

А как построить автомат Бюхи, распознающий в точности множество $Tr(\varphi)$ для LTL-формулы φ?

Начнём с примеров

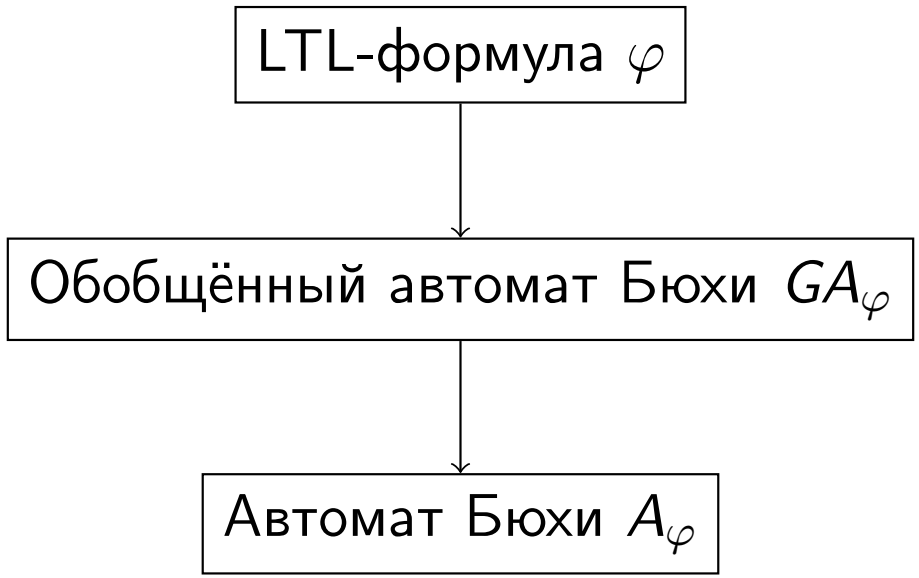
$$\mathbf{G}(a \to \mathbf{F}b)$$



А как построить такой автомат Бюхи для LTL-формулы произвольного вида?

Построение автомата Бюхи по LTL-формуле

Общая схема построения автомата Бюхи по LTL-формуле:



Обобщённые автоматы Бюхи

Обобщённый автомат Бюхи над алфавитом Σ — это система $GA = (S, S_0, \delta, \mathcal{F})$, где

- S — конечное множество состояний
- $S_0 \subseteq S$ — множество начальных состояний
- $\delta \subseteq S \times \Sigma \times S$ — отношение переходов
- $\mathcal{F} \subseteq 2^S$ — семейство множеств заключительных состояний

Отличие от *необобщённого* автомата Бюхи только в том, какие слова распознаются таким автоматом

Бесконечное слово w распознаётся обобщённым автоматом Бюхи GA, если среди трасс, порождаемых словом w, существует трасса t, такая что $inf(t) \cap F \neq \emptyset$ для любого множества F из семейства $\mathcal{F}$

Обобщённые автоматы Бюхи

Утверждение

Для любого обобщённого автомата Бюхи существует *необобщённый* автомат Бюхи, распознающий тот же язык

Доказательство.

Рассмотрим произвольный обобщённый автомат Бюхи $GA = (S, S_0, \delta, \{F_0, \ldots, F_{k-1}\})$

Требуемый автомат Бюхи $A = (S', S'_0, \delta, F)$ имеет следующее устройство:

- $S' = S \times \{0, \ldots, k-1\}$
- $S'_0 = S_0 \times \{0\}$
- $F = \{F_i \times \{i\} \mid 0 \leq i < k\}$
- $\delta'((s,i),\sigma,(s',i)) \Leftrightarrow \delta(s,\sigma,s')$ и $s \notin F_i$
- $\delta'((s,i),\sigma,(s',i+1(\mathbf{mod}\ k))) \Leftrightarrow \delta(s,\sigma,s')$ и $s \in F_i$

▼

Построение автомата Бюхи по LTL-формуле

И как же построить обобщённый автомат Бюхи, распознающий свойство φ?

Например, такой обобщённый автомат Бюхи $A = (S, S_0, \delta, \mathcal{F})$ распознаёт свойство φ:

- S — это всевозможные *внутренне согласованные предположения*, построенные на основе *замыкания Фишера-Ладнера* формулы φ
- S_0 — это всевозможные предположения H, такие что $\varphi \in H$
- Каждой формуле $\psi \mathbf{U} \chi \in [\varphi]_{FL}$ соответствует множество заключительных состояний $F_{\psi \mathbf{U} \chi}$, состоящее в точности из тех предположений H, для которых *звенит звонок* для формулы $\psi \mathbf{U} \chi$:
  - $\psi \in H$ или
  - $\mathbf{X}(\varphi \mathbf{U} \psi) \in H$
- $\delta(H_1, L, H_2) \Leftrightarrow$
  - H_1 и H_2 — *внешне согласованные предположения*
  - предположение H_1 *согласовано с атомарными высказываниями* L

Автоматный алгоритм model checking для LTL

Промежуточный итог

Для заданных модели Крипке M и LTL-формулы φ требуется проверить соотношение

$$M \models \varphi$$

Что для этого уже сделано:

- для модели M построен автомат Бюхи A_M, такой что $L(A) = Tr(M)$
- для формулы $\neg\varphi$ построен автомат Бюхи $A_{\neg\varphi}$, такой что $L(A_{\neg\varphi}) = Tr(\neg\varphi)$

Что осталось сделать:

- построить автомат $A_M \oplus A_{\neg\varphi}$, распознающий язык $L(A_M) \cap L(A_{\neg\varphi})$
- проверить, имея описание автомата, пуст ли этот язык

Пересечение автоматов Бюхи

Утверждение

Для любой пары автоматов Бюхи A_1, A_2 существует автомат Бюхи A, распознающий язык $L(A_1) \cap L(A_2)$

Доказательство.

Рассмотрим произвольные автоматы Бюхи $A_1 = (S_1, S_0^1, \delta_1, F_1)$, $A_2 = (S_2, S_0^2, \delta_2, F_2)$

Достаточно построить **обобщённый** автомат Бюхи, распознающий язык $L(A_1) \cap L(A_2)$

Требуемый обобщённый автомат Бюхи GA имеет следующее устройство:

$$GA = (S_1 \times S_2, S_0^1 \times S_0^2, \delta, \{F_1 \times S_2, S_1 \times F_2\}), \text{ где}$$

$$\delta((s_1, s_2), \sigma, (s_1', s_2')) \Leftrightarrow \delta_1(s_1, s_1') \text{ и } \delta_2(s_2, s_2')$$

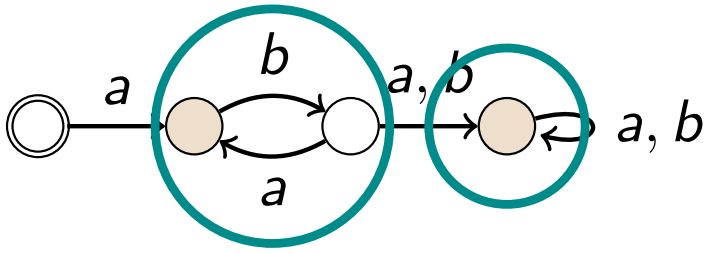
Проверка пустоты автомата Бюхи

Утверждение

Язык $L(A)$, распознаваемый автоматом Бюхи A, пуст тогда и только тогда, когда существует заключительное состояние автомата A, принадлежащее какому-либо циклу, достижимому из какого-либо начального состояния этого автомата

Доказательство.Очевидно?

Пример



Язык, распознаваемый этим автоматом, непуст

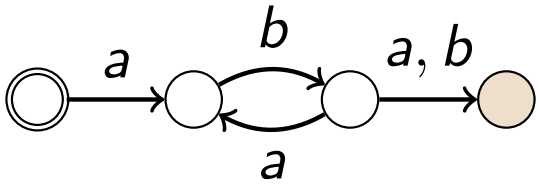
Проверка пустоты автомата Бюхи

Утверждение

Язык $L(A)$, распознаваемый автоматом Бюхи A, пуст тогда и только тогда, когда существует заключительное состояние автомата A, принадлежащее какому-либо циклу, достижимому из какого-либо начального состояния этого автомата

Доказательство.Очевидно?

Пример



Язык, распознаваемый этим автоматом, пуст

Сведение задачи model checking для LTL к проблеме пустоты для автоматов Бюхи

Итог

Пусть M — произвольная модель Крипке
и φ — произвольная LTL-формула

Тогда

$$M \models \varphi$$

$$\Leftrightarrow$$

$$L(A_M \oplus A_{\neg\varphi}) = \emptyset$$

Конец лекции 7