Математические методы верификации схем и программ

Лекторы:

Захаров Владимир Анатольевич

Подымов Владислав Васильевич

e-mail рассказчика:

valdus@yandex.ru

Осень 2016

Лекция 10

Задача model checking
для временных автоматов
и сетей временных автоматов

Эквивалентность оценок таймеров

Временные регионы
и регионы состояний

Оценка числа временных регионов

Регионная модель Крипке

Сведение MC для TCTL к MC для CTL

Напоминание

Временной автомат — это система $(L, \ell_0, \Sigma, C, I, T)$, где

- L — конечное множество состояний автомата
- $\ell_0 \in L$ — начальное состояние
- Σ — конечное множество событий
- C — конечное множество таймеров
- $I : L \to inv(C)$ — разметка состояниий инвариантами
- $T \subseteq L \times (\Sigma \cup \{\lambda\}) \times guard(C) \times 2^C \times L$ — отношение переходов
  - третья компонента перехода — предусловие
  - четвёртая компонента перехода — множество сбрасываемых таймеров

Напоминание

- $ATC(C)$ — множество всех элементарных ограничений над таймерами множества $\mathcal{C}$: **true**, **false** и всевозможные выражения вида $x \bowtie k$ и $x - y \bowtie k$, где
  - x, y — таймеры
  - k — целая неотрицательная константа
  - $\bowtie \in \{<, \leq, >, \geq, =, \neq\}$
- $inv(C)$ — множество всех формул, составленных над атомами **true**, $x < k$, $x \leq k$ и связкой $\&$
- $ALC(A)$ — множество всех выражений вида $A.\ell$, где ℓ — состояние автомата A

Напоминание

Конфигурация временного автомата $(L, \ell_0, \Sigma, C, I, T)$ — это пара (ℓ, d), где $\ell \in L$ и $d : C \to \mathbb{R}_{\geq 0}$ (d — оценка таймеров)

Начальная конфигурация автомата — это пара $(\ell_0, 0, 0, \ldots, 0)$

Шаг вычисления временного автомата — это изменение конфигурации (ℓ, d) на конфигурацию (ℓ', d') одним из двух способов:

- продвижение времени: $(\ell, d) \to (\ell', d')$
  - $\ell' = \ell$
  - существует константа $k \in \mathbb{R}_{>0}$, такая что $d' = d + k$
  - $d' \models I(\ell)$
- изменение состояния: $(\ell, d) \to (\ell', d')$
  - $(\ell, \sigma, g, \mathcal{C}, \ell') \in T$
  - $d \models g$
  - $d' = reset(d, \mathcal{C})$
  - $d' \models I(\ell')$

Напоминание

Сеть временных автоматов — это система $(C, Chan, (A_1, \ldots, A_k))$, где

- C — конечное множество общих таймеров
- $Chan$ — конечное множество общих каналов взаимодействия
- $A_i = (L^i, l_0^i, \{ch!, ch? \mid ch \in Chan\}, C, I^i, T^i)$ — временной автомат над общими таймерами и общими каналами взаимодействия $(1 \leq i \leq k)$

Напоминание

Конфигурация сети $(C, Chan, (A_1, \ldots, A_k))$,
$A_i = (L^i, \ell_0^i, \Sigma, C, I^i, T^i)$ — это система $(\ell^1, \ldots, \ell^k, d)$, где $\ell^i \in L^i$ и $d : C \to \mathbb{R}_{\geq 0}$

Начальная конфигурация сети имеет вид $(\ell_0^1, \ldots, \ell_0^k, 0, 0, \ldots, 0)$

Шаг вычисления сети — это изменение конфигурации $(\ell_1, \ldots, \ell_k, d)$ на конфигурацию $(\ell_1', \ldots, \ell_k', d')$ одним из трёх способов:

1. продвижение времени:
   - $\ell_1' = \ell_1, \ldots, \ell_k' = \ell_k$
   - существует положительная действительная константа D, такая что $d' = d + D$
   - $d' \models I^1(\ell_1) \,\&\, \ldots \,\&\, I^k(\ell_k)$

Напоминание

Конфигурация сети $(C, Chan, (A_1, \ldots, A_k))$,
$A_i = (L^i, \ell^i_0, \Sigma, C, I^i, T^i)$ — это система $(\ell^1, \ldots, \ell^k, d)$, где $\ell^i \in L^i$ и $d : C \to \mathbb{R}_{\geq 0}$

Начальная конфигурация сети имеет вид $(\ell^1_0, \ldots, \ell^k_0, 0, 0, \ldots, 0)$

Шаг вычисления сети — это изменение конфигурации $(\ell_1, \ldots, \ell_k, d)$ на конфигурацию $(\ell'_1, \ldots, \ell'_k, d')$ одним из трёх способов:

2. асинхронное изменение состояния автомата A_i:
   - $\ell'_p = \ell_p$ для $p \neq i$
   - $(\ell_i, \lambda, g, \mathcal{C}, \ell'_i) \in T^i$
   - $d \models g$
   - $d = reset(d', \mathcal{C})$
   - $d' \models I^i(\ell'_i)$

Напоминание

Конфигурация сети $(C, Chan, (A_1, \dots, A_k))$,
$A_i = (L^i, \ell_0^i, \Sigma, C, I^i, T^i)$ — это система $(\ell^1, \dots, \ell^k, d)$, где $\ell^i \in L^i$ и $d : C \to \mathbb{R}_{\geq 0}$

Начальная конфигурация сети имеет вид $(\ell_0^1, \dots, \ell_0^k, 0, 0, \dots, 0)$

Шаг вычисления сети — это изменение конфигурации $(\ell_1, \dots, \ell_k, d)$ на конфигурацию $(\ell'_1, \dots, \ell'_k, d')$ одним из трёх способов:

3. синхронное изменение состояний автоматов A_i, A_j:
   - $\ell'_p = \ell_p$ для $p \neq i$, $p \neq j$
   - $(\ell_i, c!, g', \mathcal{C}', \ell'_i) \in T^i$
   - $(\ell_j, c?, g'', \mathcal{C}'', \ell'_j) \in T^j$
   - $d \models g' \,\&\, g''$
   - $d' = reset(d, \mathcal{C}' \cup \mathcal{C}'')$
   - $d' \models I^i(\ell'_i) \,\&\, I^j(\ell'_j)$

Напоминание

Бесконечная модель Крипке $M(A)$ [$M(N)$] временного автомата A [сети временных автоматов N] определяется так:

- состояние модели — это конфигурация
- начальное состояние — это начальная конфигурация
- переходами связаны пары конфигураций, образующих шаг вычисления
- состояния модели размечены истинными в этих состояниях атомарными высказываниями множеств $ATC(C)$, $ALC(A)$, где C — таймеры автомата A [$ATC(C)$, $ALC(A_i)$, где C — общие таймеры сети N и A_i — автоматы сети]

Полагаем, что “справедливостью” исключаются конвергентные вычисления: бесконечное продвижение времени без смен состояний и с конечной верхней временной границей

Напоминание

Синтаксис формул логики TCTL (Timed CTL) совпадает с синтаксисом формул логики CTL **без оператора X** над множеством атомарных высказываний модели Крипке автомата или сети

Семантика TCTL-формул отличается тем, что смысл темпоральных операторов адаптирован к работе системы в реальном времени в условиях дискретной модели Крипке

$M \models_{TCTL} \varphi$: TCTL-формула φ выполнена в модели Крипке M

Задача model checking для (сетей) временных автоматов

Задача model checking для временных автоматов (MCTA)

Для временного автомата A и TCTL-формулы φ проверить справедливость соотношения

$$M(A) \models_{TCTL} \varphi$$

Задача model checking для сетей временных автоматов (MCNTA)

Для сети временных автоматов N и TCTL-формулы φ проверить справедливость соотношения

$$M(N) \models_{TCTL} \varphi$$

А можно ли *более сложную* задачу MCNTA свести к *более простой* задаче MCTA?

Трансляция сети временных автоматов во временной автомат

Рассмотрим сеть временных автоматов
$N = (C, Chan, (A_1, \dots, A_m))$, где
$A_i = (L^i, \ell_0^i, \{ch!, ch? \mid ch \in Chan\}, C, I^i, T^i)$

Построим по ней такой временной автомат
$A(N) = (L, \ell_0, \emptyset, C, I, T)$:

- $L = L^1 \times \dots \times L^m$
- $\ell_0 = (\ell_0^1, \dots, \ell_0^m)$
- $I(\ell_1, \dots, \ell_m) = I^1(\ell_1) \,\&\, \dots \,\&\, I^m(\ell_m)$
- $((\ell_1, \dots, \ell_m), \lambda, g, \mathcal{C}, (\ell'_1, \dots, \ell'_m)) \in T$ тогда и только тогда, когда верно одно из условий:
  - для какого-либо автомата A_i верно $(\ell_i, \lambda, g, \mathcal{C}, \ell'_i) \in T^i$, и $\ell_p = \ell'_p$ при $p \neq i$
  - для каких-либо автоматов A_i, A_j, $i \neq j$, верно $(\ell_i, c!, g_1, \mathcal{C}_1, \ell'_i) \in T^i$ и $(\ell_j, c?, g_2, \mathcal{C}_2, \ell'_j) \in T_j$; $\ell_p = \ell'_p$ при $p \notin \{i, j\}$; $g = g_1 \,\&\, g_2$; $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$

Трансляция сети временных автоматов во временной автомат

Переразметим модель Крипке $M(A(N))$ автомата $A(N)$ следующим образом: метку $A.(\ell_1, \dots, \ell_m)$ заменим на метки $A.\ell_1, \dots, A.\ell_m$

Пусть в результате получена бесконечная модель $\widetilde{M}(A(N))$

Утверждение. $M(N) = \widetilde{M}(A(N))$

Упражнение. Строго докажите это утверждение

Следствие. $M(N) \models \varphi \quad \Leftrightarrow \quad \widetilde{M}(A(N)) \models \varphi$

Так задачу MCNTA можно свести к задаче MCTA

Трудность задачи МСТА

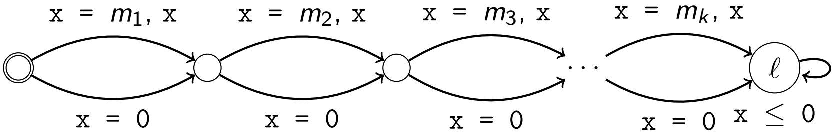
Задача о сумме подмножеств (СП)

Даны натуральное число N и множество натуральных чисел $S = \{m_1, \ldots, m_k\}$

Выяснить, существует ли подмножество множества S, такое что сумма чисел этого подмножества есть N

Утверждение. **Задача о сумме подмножеств NP-полна**

Рассмотрим такой временной автомат A с таймерами x, y:



Утверждение. **Задача СП имеет решение** $\Leftrightarrow$

$$M(A) \models \mathbf{EF}(A.\ell \,\&\, y = N)$$

И что же это говорит о трудности задачи МСТА?

Сжатие модели Крипке

Алгоритмы проверки CTL-формул на моделях Крипке не применимы напрямую к решению задачи MCTA:

- модель $M(A)$ временного автомата A в общем случае бесконечна
- семантика TCTL отличается от семантики CTL

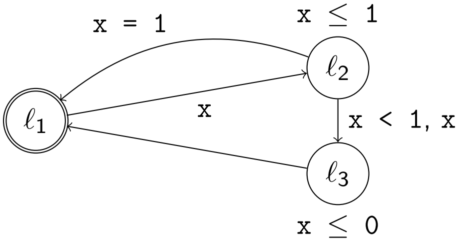
Попытаемся добиться того, чтобы алгоритмы стали применимы

Для этого "сожмём" бесконечную модель временного автомата в конечную модель Крипке так, чтобы исходная TCTL-формула оказалась равновыполнима с синтаксически совпадающей конечной CTL-формулой

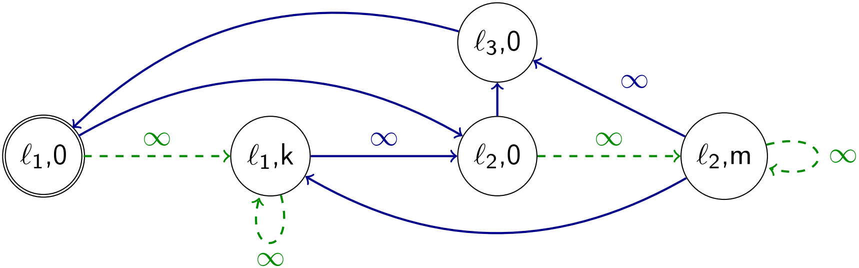
Для этого попытаемся объединить множество значений таймеров в конечное число классов

Сжатие модели Крипке

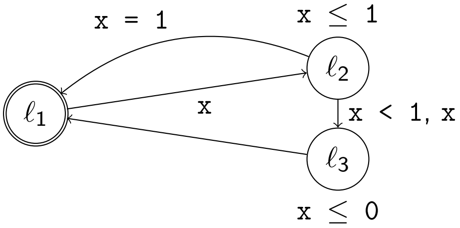
Пример



Бесконечная модель Крипке для этого автомата:



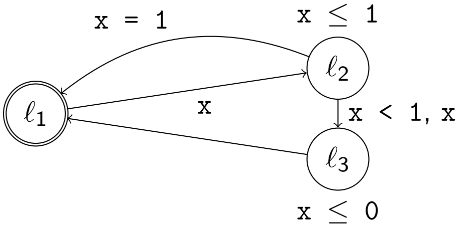
Сжатие модели Крипке



А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель **сжатием***)

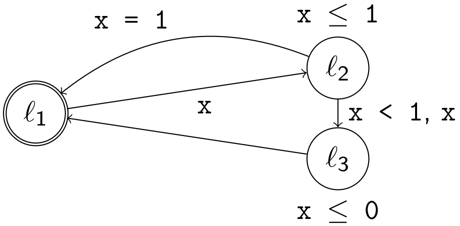
Сжатие модели Крипке



А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

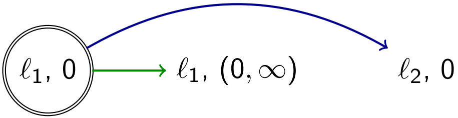
(*пока что будем называть такую модель* ***сжатием***)

Сжатие модели Крипке

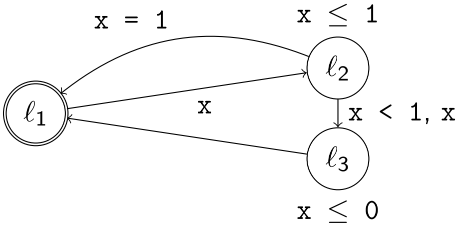


А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель* ***сжатием***)

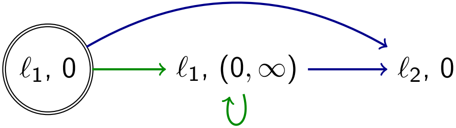


Сжатие модели Крипке

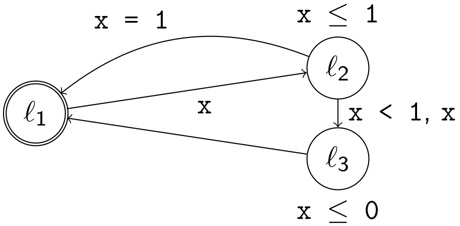


А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель* ***сжатием***)

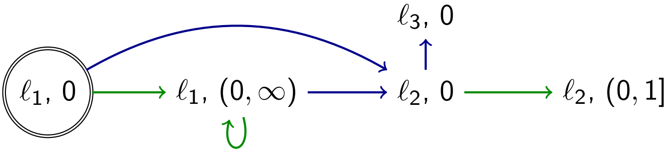


Сжатие модели Крипке

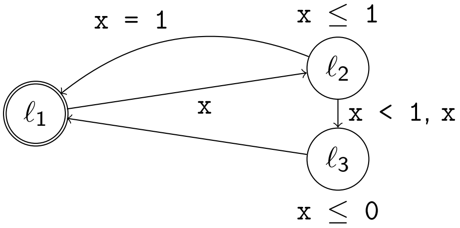


А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель* ***сжатием***)

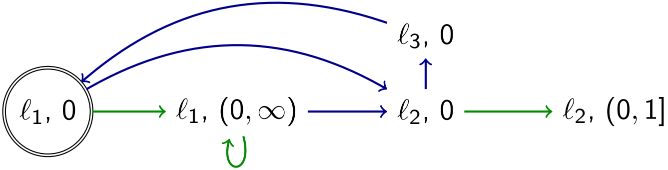


Сжатие модели Крипке

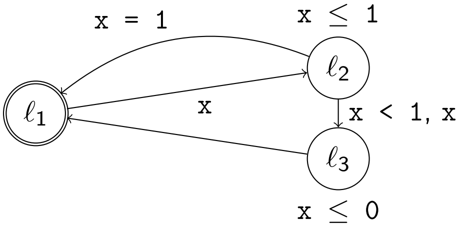


А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель* ***сжатием***)

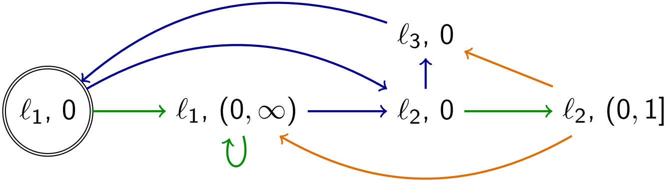


Сжатие модели Крипке

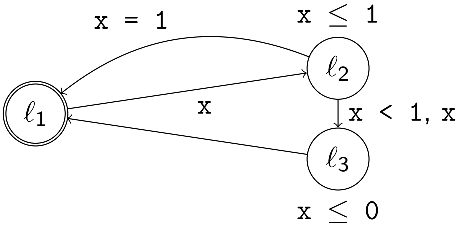


А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель* ***сжатием***)

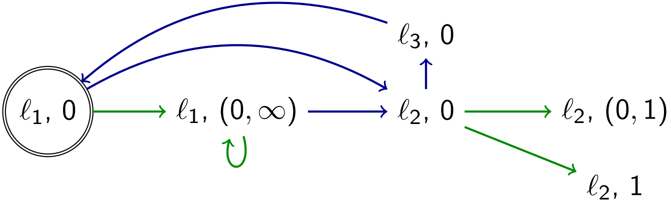


Сжатие модели Крипке

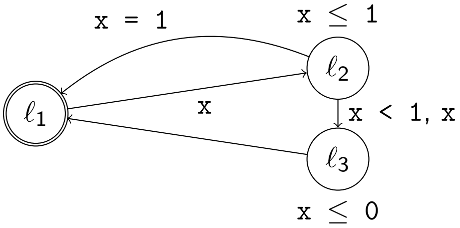


А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель* ***сжатием***)

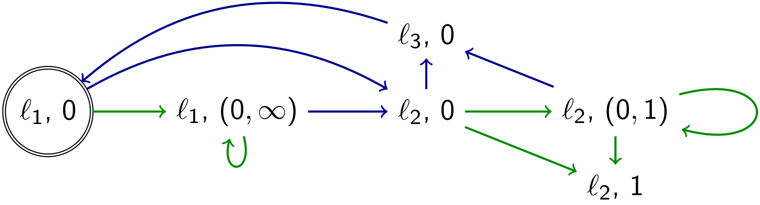


Сжатие модели Крипке

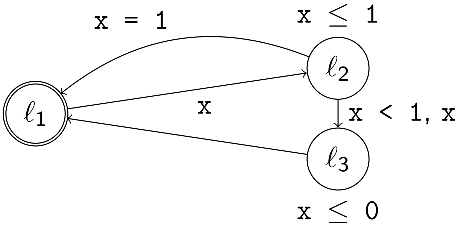


А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель* ***сжатием***)

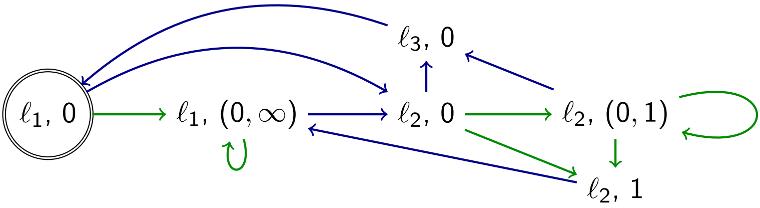


Сжатие модели Крипке

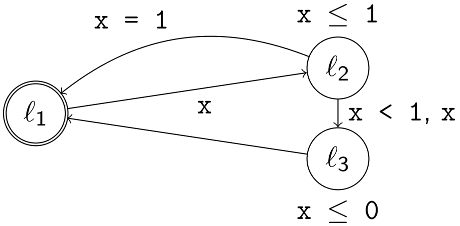


А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель* ***сжатием***)

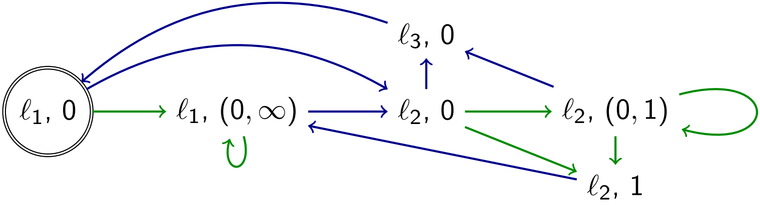


Сжатие модели Крипке



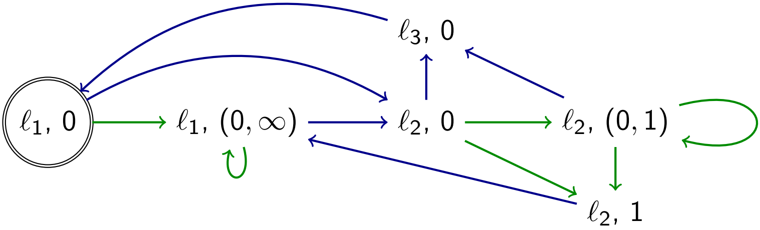
А можно ли для этого автомата построить конечную модель Крипке, достаточную для решения задачи MCTA *хотя бы* для конкретного автомата и конкретной формулы?

(*пока что будем называть такую модель* ***сжатием***)



Подходит ли такая модель для наших целей?

Сжатие модели Крипке



$\mathbf{AG}(A.\ell_2 \to \mathbf{A}(x \leq 1 \mathbf{U} A.\ell_1))$: выполнена в бесконечной и конечной моделях

$\mathbf{AG}(A.\ell_2 \,\&\, x = 1 \to \mathbf{A}(x \geq 1 \mathbf{U} A.\ell_2))$: выполнена в бесконечной модели; а в конечной?

А как в этой модели разметить состояния атомарными высказываниями?

Сжатие модели Крипке

Если размечать сжатие атомарными высказываниями "по-честному", то возникает неоднозначность в разметке элементарными ограничениями таймеров

Например, если $x \in (0, \infty)$, то неясно, верно ли в состоянии с этим интервалом ограничение $x \geq 1$

При этом в некоторых состояниях необходимо знать, верно ли $x \geq 1$, **если проверяется формула**

$$\mathbf{AG}(A.\ell_2 \,\&\, x = 1 \rightarrow \mathbf{A}(x \geq 1 \mathbf{U} A.\ell_2))$$

А значение ограничения $x \geq 2$ **абсолютно неважно**

И какие же элементарные ограничения нам важны при построении сжатия?

Те, которые содержатся в проверяемой формуле φ

Множество таких ограничений будем обозначать записью $ATC(\varphi)$

Осталось понять, какой именно вид должны иметь состояния сжатия

Эквивалентность оценок таймеров, регионы

Если временной автомат содержит n таймеров, то всевозможные значения таймеров образуют пространство R^n

Попытаемся разбить это пространство на конечное число классов эквивалентности так, чтобы можно было

- однозначно пометить каждое состояние сжатия высказываниями из $ALC(A) \cup ATC(\varphi)$
- гарантировать, что дуги, исходящие из каждого состояния сжатия, в точности соответствуют дугам, исходящим из каждого состояния исходной бесконечной модели, описываемого выбранным состоянием сжатия

Когда такое отношение будет построено, достаточно будет объявить пару, состоящую из состояния автомата и класса эквивалентности таймеров, состоянием сжатия и естественным образом расставить переходы модели

Эквивалентность оценок таймеров, регионы

Далее считаем заданными временной автомат $A = (L, \ell_0, \emptyset, C, I, T)$ и TCTL-формулу φ, выполнимость которой проверяется в модели $M(A)$

Для простоты технических выкладок считаем, что ни в A, ни в φ не встречается выражений вида $x - y \bowtie k$

Предположим, что в A или φ встречается выражение $x \bowtie k$, $\bowtie \in \{<, \leq, >, \geq, =, \neq\}$

Чтобы знать, верно ли это выражение при оценке таймеров d, достаточно знать

- целую часть значения $d(x)$: $\lfloor d(x) \rfloor$
- имеет ли значение $d(x)$ ненулевую дробную часть $frac(d(x))$

Эквивалентность оценок таймеров, регионы

Можно попытаться ввести эквивалентность оценок таймеров так: оценки d, d' эквивалентны, если для любого таймера $x \in C$ верно

- $\lfloor d(x) \rfloor = \lfloor d'(x) \rfloor$ и
- $frac(d(x)) = 0 \quad \Leftrightarrow \quad frac(d(x)) = 0$

Тогда для эквивалентных оценок d, d' будет верно:

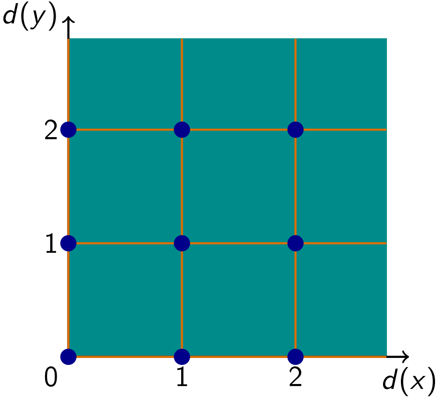
$$d(x) \bowtie k \quad \Leftrightarrow \quad d'(x) \bowtie k$$

Эквивалентность оценок таймеров, регионы

Пример

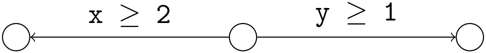
Предположим, что автомат содержит два таймера: x и y
Классы *введённой только что* эквивалентности этих таймеров будут выглядеть так:

(*один класс — связный сегмент одного цвета*)



Эквивалентность оценок таймеров, регионы

Рассмотрим такую пару переходов автомата:



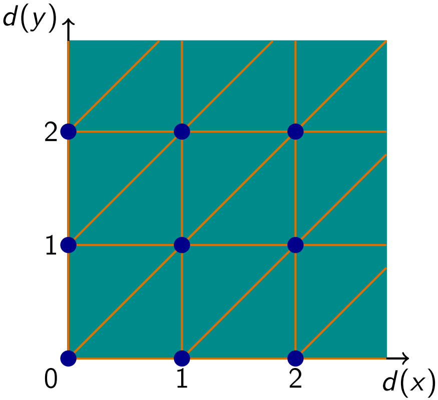
Предположим, что текущее состояние сжатия — центральное, и что текущий класс эквивалентности сжатия — $\{(x, y) \mid 1 < x < 2, 0 < y < 1\}$

В зависимости от выбора представителя класса эквивалентности при продвижении времени могут быть получены два *существенно различных* состояния:

- в одном возможно выполнения перехода влево, но невозможно выполнение перехода вправо
- в другом возможно выполнение перехода вправо, но невозможно выполнение перехода влево

В исходной модели Крипке такая ситуация невозможна, а значит, введённое отношение эквивалентности не подходит для сжатия

Эквивалентность оценок таймеров, регионы

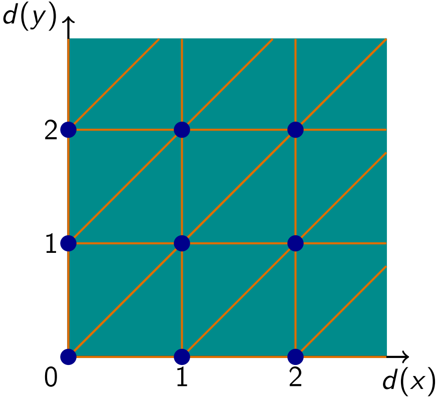


Добавим в список условий, при выполнении которых оценки таймеров d, d' признаются эквивалентными, такое:

- для любой пары таймеров x, y верно:
  $frac(d(x)) \leq frac(d(y)) \quad \Leftrightarrow \quad frac(d'(x)) \leq frac(d'(y))$

Это условие исключает “плохую” ситуацию, описанную выше

Эквивалентность оценок таймеров, регионы



Такое отношение эквивалентности всё равно не подходит для описания сжатия, так как **число классов эквивалентности бесконечно**

Попытаемся объединить некоторые классы эквивалентности так, чтобы сделать число классов конечным

Эквивалентность оценок таймеров, регионы

Пусть k_x — максимальная константа, встречающаяся в выражениях A и φ вида $x \bowtie k$

Тогда если $d(x) > k_x$, то каким бы ни было значение $d(x)$, все высказывания $x \bowtie k$ в A и φ будут иметь одно и то же значение

Объединим классы эквивалентности так: если d и d' отличаются только значением x и при этом $d(x) > k_x$ и $d'(x) > k_x$, то объявим d и d' эквивалентными

Теперь можно подвести итог рассуждений и сформулировать понятие эквивалентности оценок таймеров

Эквивалентность оценок таймеров, регионы

Оценки таймеров d, d' эквивалентны, если выполнены следующие условия:

- для любого таймера x верно: $d(x) > k_x \quad \Leftrightarrow \quad d'(x) > k_x$
- для любых таймеров x, y, таких что $d(x) \leq k_x$ и $d(y) \leq k_y$, верно:
  - $\lfloor d(x) \rfloor = \lfloor d'(x) \rfloor$
  - $frac(d(x)) = frac(d'(x))$
  - $frac(d(x)) \leq frac(d(y)) \quad \Leftrightarrow \quad frac(d'(x)) \leq frac(d'(y))$

Регионами оценок будем называть классы эквивалентности оценок таймеров

$[d]$ — это регион, которому принадлежит d

Регионами конфигураций будем называть пары (ℓ, r), где $\ell \in L$ и r — это регион оценок

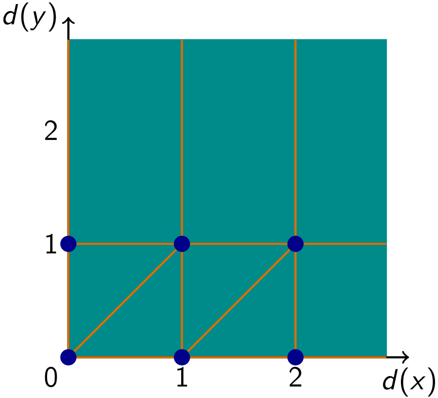
Если $d(x) > k_x$, то регион $[d]$ будем называть открытым для x, в противном случае — закрытым для x

Эквивалентность оценок таймеров, регионы

Пример

Пусть автомат A содержит два таймера: x и y — и пусть $k_x = 2$ и $k_y = 1$

Тогда регионы оценок будут выглядеть так:



Оценка числа регионов

Утверждение

Число N попарно различных регионов оценок для временного автомата $A = (L, \ell_0, C, I, T)$ и формулы φ, таких что $k_x \geq 1$ для каждого таймера $x \in C$, оценивается снизу и сверху такими константами:

$$|C|! \cdot \prod_{x \in C} k_x \leq N \leq |C|! \cdot 2^{|C|-1} \cdot \prod_{x \in C} (2k_x + 2)$$

Доказательство.

Откуда берётся

- $\prod\limits_{x \in C} k_x$: отношение эквивалентности содержит столько единичных кубов размерности $|C|$, таких что регионы внутри этих кубов закрыты для всех таймеров
- $|C|!$: столькими способами можно определить порядок дробных частей таймеров региона

Оценка числа регионов

Утверждение

Число N попарно различных регионов оценок для временного автомата $A = (L, \ell_0, C, I, T)$ и формулы φ, таких что $k_x \geq 1$ для каждого таймера $x \in C$, оценивается снизу и сверху такими константами:

$$|C|! \cdot \prod_{x \in C} k_x \leq N \leq |C|! \cdot 2^{|C|-1} \cdot \prod_{x \in C} (2k_x + 2)$$

Доказательство.

Откуда берётся

- $2^{|C|-1}$: столькими способами для каждого порядка можно выбрать, какие из соседних в порядке дробных частей равны
- $2k_x + 2$: столькими способами можно выбрать диапазон допустимых значений таймера в регионе ▼

Операции над регионами

Чтобы коротко описать сжатие, определим, как изменяются регионы оценок при продвижении времени и сбросе таймеров

Рассмотрим регион r, подмножество таймеров $\mathcal{C}$ и константу k

Регион $reset(r, \mathcal{C})$ состоит из всех оценок $reset(d, \mathcal{C})$, где $d \in r$

Открытый регион — это регион, открытый для всех таймеров

Продвижение региона r ($succ(r)$) определяется так:

- если r — открытый регион, то значение $succ(r) = r$
- иначе $succ(r)$ — регион, **отличный от** r и такой что для любой содержащейся в нём оценки d' верно: $d' = d + k$, где $d \in r$, и любая оценка вида $d + k'$, $0 \leq k' \leq k$, содержится либо в r, либо в $succ(r)$

Операции над регионами

Утверждение

Если элементарное ограничение a, содержащееся в автомате A или формуле φ, верно для какой-либо оценки d региона r, построенного для A и φ, то оно верно для всех оценок этого региона

Доказательство.Очевидно?

Отсылая к этому утверждению, будем говорить, что формула φ над элементарными ограничениями, содержащимися в A и φ, и булевыми связками верна в регионе r ($r \models \varphi$), если она верна для любой оценки этого региона, и неверна в регионе ($r \not\models \varphi$) иначе

Регионная модель Крипке

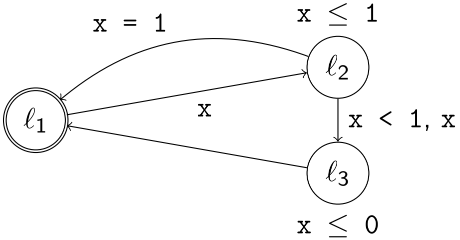
Регионная модель Крипке $M_r(A, \varphi)$ для автомата $A = (L, \ell_0, \emptyset, I, T)$ и TCTL-формулы φ определяется так:

- состояния модели — это всевозможные регионы конфигураций
- начальное состояние модели состоит из начального состояния автомата и региона $[(0, \ldots, 0)]$
- модель содержит следующие переходы:
  - если автомат содержит переход $(\ell, \lambda, g, \mathcal{C}, \ell')$, $r \models g$ и $reset(r, \mathcal{C}) \models I(\ell')$, то модель содержит переход $(\ell, r) \to (\ell', reset(r, \mathcal{C}))$
  - если $r \models I(\ell)$ и $succ(r) \models I(\ell)$, то модель содержит переход $(\ell, r) \to (\ell, succ(r))$
- состояние модели (ℓ, r) помечено всеми высказываниями из $ALC(A) \cup ATC(\varphi)$, верными в регионе r

Регионная модель Крипке — это и есть то, что ранее условно называлось *сжатием*

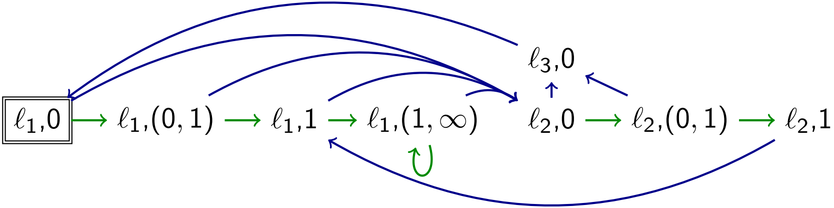
Регионная модель Крипке

Пример



φ: $\mathbf{AG}(A.\ell_2 \,\&\, x = 1 \to \mathbf{A}(x \geq 1 \mathbf{U} A.\ell_2))$

Достижимый фрагмент регионной модели Крипке для этих автомата и формулы выглядит так:



Сведение MCTA к MC моделей Крипке относительно CTL

Теорема

Для любых временного автомата A и TCTL-формулы φ, если из модели $M(A)$ "справедливостью" исключены конвергентные вычисления, то

$$M(A) \models_{TCTL} \varphi \quad \Leftrightarrow \quad M_r(A, \varphi) \models \varphi$$

Доказательство (*идея*).

Попробуем построить отношение состояний $M(A)$ и $M_r(A)$, как можно более похожее на *бисимуляцию*

Настолько похожее, чтобы идеи равновыполнимости формул в бисимуляционно эквивалентных моделях были применимы и к этому отношению

Состоянию (ℓ, d) модели $M(A)$ поставим в соответствие регион $[(\ell, d)] = (\ell, [d])$

Сведение MCTA
к MC моделей Крипке относительно CTL

Теорема

Для любых временного автомата A и TCTL-формулы φ, если из модели $M(A)$ "справедливостью" исключены конвергентные вычисления, то

$$M(A) \models_{TCTL} \varphi \quad \Leftrightarrow \quad M_r(A, \varphi) \models \varphi$$

Доказательство (*идея*).

Какие свойства делают отношение [] похожим на бисимуляцию?

Какими бы ни были переход $(\ell, r) \to (\ell', r')$ в модели $M_r(A)$ и []-прообраз (ℓ, d) состояния (ℓ, r), модель $M(A)$ содержит переход $(\ell, d) \to (\ell', d')$, где $d' \in r'$

Обратное *почти* верно: если переход $(\ell, d) \to (\ell', d')$ в $M(A)$ —

- изменение состояния, то модель $M_r(A)$ содержит переход $(\ell, [d]) \to (\ell', [d'])$
- продвижение времени, то всё хуже: модель $M_r(A)$ может содержать 0, 1 или несколько соответствующих переходов

Сведение MCTA
к MC моделей Крипке относительно CTL

Теорема

Для любых временного автомата A и TCTL-формулы φ, если из модели $M(A)$ “справедливостью” исключены конвергентные вычисления, то

$$M(A) \models_{TCTL} \varphi \quad \Leftrightarrow \quad M_r(A, \varphi) \models \varphi$$

Доказательство (*идея*).

- продвижение времени, то всё хуже: модель $M_r(A)$ может содержать 0, 1 или несколько соответствующих переходов

Один переход — если $[d'] = succ([d])$

Ни одного перехода, если $[d] = [d']$ и это не открытый регион

Несколько переходов через регионы $[d], succ([d]), succ(succ([d])), \ldots, [d']$ в остальных случаях

“Несколько переходов” — это не страшно, а вот “ни одного перехода” — это плохое свойство

Сведение MCTA
к MC моделей Крипке относительно CTL

Теорема

Для любых временного автомата A и TCTL-формулы φ, если из модели $M(A)$ “справедливостью” исключены конвергентные вычисления, то

$$M(A) \models_{TCTL} \varphi \quad \Leftrightarrow \quad M_r(A, \varphi) \models \varphi$$

Доказательство (*идея*).

- продвижение времени, то всё хуже: модель $M_r(A)$ может содержать 0, 1 или несколько соответствующих переходов

Если $[d] = [d']$, можно “заглянуть вперёд” в вычисление после $(\ell, d) \rightarrow (\ell, d')$:

- если с этого момента постоянно продвигается время, то *так как это вычисление дивергентно*, в модели $M_r(A)$ есть путь, в котором аналогичным образом продвигается время

Сведение MCTA к MC моделей Крипке относительно CTL

Теорема

Для любых временного автомата A и TCTL-формулы φ, если из модели $M(A)$ “справедливостью” исключены конвергентные вычисления, то

$$M(A) \models_{TCTL} \varphi \quad \Leftrightarrow \quad M_r(A, \varphi) \models \varphi$$

Доказательство (*идея*).

- продвижение времени, то всё хуже: модель $M_r(A)$ может содержать 0, 1 или несколько соответствующих переходов

Если $[d] = [d']$, можно “заглянуть вперёд” в вычисление после $(\ell, d) \to (\ell, d')$:

- если дальше в вычислении несколько раз продвигается время и затем изменяется состояние, то в $M_r(A)$ можно некоторым числом переходов поднять время до того же уровня и точно так же изменить состояние

Сведение MCTA к MC моделей Крипке относительно CTL

Теорема

Для любых временного автомата A и TCTL-формулы φ, если из модели $M(A)$ “справедливостью” исключены конвергентные вычисления, то

$$M(A) \models_{TCTL} \varphi \quad \Leftrightarrow \quad M_r(A, \varphi) \models \varphi$$

Доказательство (*идея*).

Описанное соответствие одного перехода $M(A)$ произвольному числу переходов в $M_r(A)$ делает отношение [] достаточно похожим на бисимуляцию, чтобы утверждать справедливость теоремы ▼

А где в доказательстве переход от семантики TCTL к семантике CTL?

Сведение MCTA к MC моделей Крипке относительно CTL

Итог:

Для любого временного автомата A и любой формулы φ логики TCTL

$$M(A) \models_{TCTL} \varphi \quad \Leftrightarrow \quad M_r(A) \models \varphi$$

Конец лекции 10