Математические методы верификации схем и программ

mk.cs.msu.ru → Лекционные курсы
→ Математические методы верификации схем и программ

Блок 30

Системы реального времени (СРВ)
Временны́е автоматы

Лектор:
Подымов Владислав Васильевич
E-mail:
valdus@yandex.ru

ВМК МГУ, 2024/2025, осенний семестр

Системы реального времени

Система реального времени (СРВ) — это система, поведение которой существенно зависит не только от порядка выполнения действий и изменения состояний, но и от того, **за какое время** выполняются действия и изменяются состояния

Для СРВ характерны директивные сроки выполнения действий компонентами: интервалы (*действительных чисел*), которым должна принадлежать длительность выполнения действий

Но в СРВ, даже разработанной в том или ином смысле «правильно», не всегда соблюдаются директивные сроки, хотя и «должны»

Для сравнения: если «правильным» считать студента, который в итоге успешно выпускается, то по уставу университета такой студент должен посещать лекции, но это не значит, что он действительно их посещает

Системы реального времени

В зависимости от того, к каким последствиям приводит несоблюдение директивных сроков, СРВ принято причислять к одному из двух классов:

- В мягких СРВ последствия хотя и нежелательны (ухудшают качество выполнения системы), но в целом допустимы
- В жёстких СРВ сорванный директивный срок считается недопустимым, приводящим к фатальному сбою с бессмысленностью продолжения выполнения

Системы реального времени

Примеры сорванных сроков в мягких СРВ:

- Поспал на два часа меньше нормы $\Rightarrow$ будешь вялым, но если не злоупотреблять, то жить будешь
- Почта задержалась на год $\Rightarrow$ печально, но все к этому привыкли
- Процесс долго освобождал память $\Rightarrow$ операционная система «подвиснет», но потом восстановится

Примеры сорванных сроков в жёстких СРВ:

- Парашют раскрылся на минуту позже документации $\Rightarrow$ смерть
- Схемные сигналы не стабилизировались в процессоре за такт $\Rightarrow$ процессор отправляется на свалку
- Светофор стал зелёным раньше положенного $\Rightarrow$ авария

Далее (*так или иначе, явно или неявно*) будут рассматриваться **только жёсткие** СРВ

Системы реального времени

Пример

Рассмотрим СРВ $\mathfrak{S}$, предназначенную для распознавания одинарного и двойного нажатия кнопки мыши:

- В $\mathfrak{S}$ выполняются (происходят в окружении и порождаются системой) действия
  - *click*: нажата кнопка мыши
  - *single*: произошло одинарное нажатие
  - *double*: произошло двойное нажатие
- Если в режиме ожидания нажатия выполнилось *click*, то через *единицу времени* $\mathfrak{S}$ принимает решение о том, какое нажатие произошло, одинарное или двойное, и порождает соответствующее действие
  - Если после первого *click* до вынесения решения ещё раз выполнилось *click*, то нажатие двойное
  - Иначе — одинарное

Системы реального времени

Пример

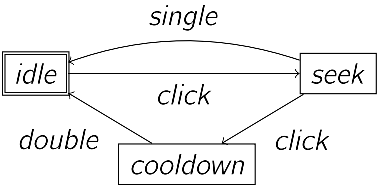
Начнём формализацию $\mathfrak{S}$ с системы переходов, отвечающей всем возможностям выполнения действий без учёта директивных сроков

Начальное состояние (*idle*) отвечает режиму ожидания первого *click*

По первому *click* перейдём в состояние *seek* ожидания второго *click*

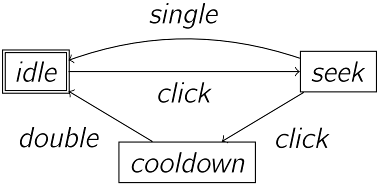
Если второе *click* не обнаружено в заданный срок, то выносится решение об одинарном нажатии: перейдём в *idle*, породив действие *single*

Если второе *click* обнаружено, то перейдём в режим *cooldown*, и через некоторое время вынесем решение о двойном нажатии: перейдём в *idle*, породив действие *double*



Системы реального времени

Пример



Чтобы следить за директивными сроками, добавим в модель часы-секундомеры:

- Значение часов — это действительное число, показывающее, сколько времени прошло с их последнего сброса
- Сбрасывать часы будем по желанию при выполнении переходов

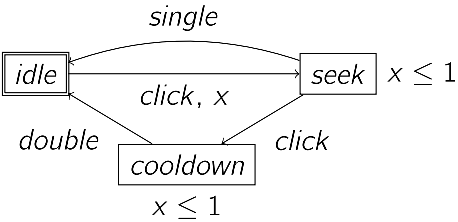
В этом примере достаточно одних часов x

При выполнении перехода $idle \to seek$ сбросим часы x, чтобы следить за тем, не пора ли выносить решение о нажатии

Чтобы обозначить сброс часов на переходе, пометим переход этими часами

Системы реального времени

Пример



В некоторых состояниях $\mathfrak{S}$ не может находиться сколь угодно долго

В состояниях *seek* и *cooldown* значение 1 часов x означает, что пришла пора выносить решение о нажатии и переходить в *idle*

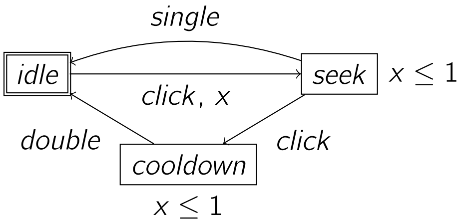
Значит, в этих состояниях значение x не может быть больше 1, то есть верно $x \leq 1$

Пометим *seek* и *cooldown* этим неравенством

Ограничение на длительность нахождение в состоянии называется инвариантом

Системы реального времени

Пример



Для каждого значения x каждый переход либо открыт (может быть выполнен), либо закрыт (не может быть выполнен)

Переходы *seek* $\to$ *idle* и *cooldown* $\to$ *idle* открыты $\Leftrightarrow x = 1$

Переход *seek* $\to$ *cooldown* открыт $\Leftrightarrow x < 1$

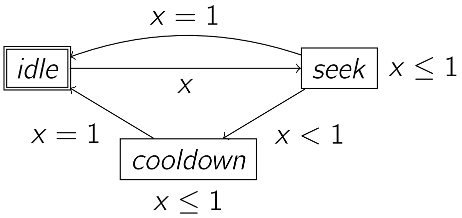
Остальные переходы открыты всегда

Пометим переходы выражениями, истинность которых равносильна открытости этих переходов

Такие выражения называются предусловиями переходов (англ. guard; иногда переводится как «охрана», «охранник» и «страж»)

Системы реального времени

Пример



Теперь «забудем» (*по крайней мере на время*) о действиях, ограничившись только тем, какими свойствами обладают состояния системы

(*По аналогии с тем, как «забываются» действия системы переходов, чтобы из неё получилась модель Крипке*)

В результате получилась модель, похожая на модель Крипке, но содержащая часы (реального времени) и предназначенная для моделирования СРВ: временной автомат

Временны́е автоматы: временны́е ограничения

Синтаксис временны́х ограничений над множеством часов $\mathcal{C}$ задаётся следующей БНФ:

$$\begin{array}{lcl} g & ::= & ag \mid (g \,\&\, g) \mid (\neg g), \\ ag & ::= & \mathbb{t} \mid (x < k) \mid (x \le k), \end{array}$$

где g — временное ограничение, ag — атомарное временное ограничение, $x \in \mathcal{C}$ и $k \in \mathbb{N}_0$

$\mathbb{R}$, $\mathbb{R}_{\ge 0}$ и $\mathbb{R}_{>0}$ — так будем обозначать множества действительных, неотрицательных действительных и положительных действительных чисел соответственно

Временны́е автоматы: временны́е ограничения

Оценка часов множества $\mathcal{C}$ — это отображение вида $\nu : \mathcal{C} \to \mathbb{R}_{\geq 0}$

Выполнимость временно́го ограничения g на оценке часов ν ($\nu \models g$) определяется естественным образом:

- Всегда верно $\nu \models \mathbb{t}$
- $\nu \models (x < k) \quad \Leftrightarrow \quad \nu(x) < k$
- $\nu \models (x \leq k) \quad \Leftrightarrow \quad \nu(x) \leq k$
- $\nu \models (g_1 \,\&\, g_2) \quad \Leftrightarrow \quad \nu \models g_1$ и $\nu \models g_2$
- $\nu \models (\neg g) \quad \Leftrightarrow \quad \nu \not\models g$

Временно́е ограничение будем называть инвариантным, если в нём не содержится $\neg$

$CC(\mathcal{C})$, $AC(\mathcal{C})$ и $IC(\mathcal{C})$ — так будем обозначать множества временны́х ограничений над $\mathcal{C}$ — всевозможных, атомарных и инвариантных соответственно

Временны́е автоматы: временны́е ограничения

В синтаксисе временны́х ограничений будут использоваться и другие булевы операции и арифметические отношения, расценивающиеся как естественные сокращения:

- $\mathbb{f} \quad = \quad \neg\mathbb{t}$
- $g_1 \vee g_2 = \neg((\neg g_1)\,\&(\neg g_2))$
- $g_1 \to g_2 = (\neg g_1) \vee g_2$
- $x \geq k = \neg(x < k)$
- $x > k = \neg(x \leq k)$
- $x = k = (x \leq k)\,\&(x \geq k)$
- $x \neq k = \neg(x = k)$

Во всех этих сокращениях используется $\neg$, а значит, их **нельзя** использовать в инвариантных ограничениях

В записи временны́х ограничений будем опускать скобки согласно обычным приоритетам булевых операций

Временны́е автоматы: синтаксис

Временной автомат над конечными множествами атомарных высказываний AP и часов $\mathcal{C}$ — это система $\mathcal{A} = (S, s_0, \mathcal{I}, T, L)$, где:

- S — конечное множество состояний
- $s_0 \in S$ — начальное состояние
- $\mathcal{I} : S \to IC(\mathcal{C})$ — разметка состояний инвариантами
- $T \subseteq S \times CC(\mathcal{C}) \times 2^{\mathcal{C}} \times S$ — отношение переходов
- $L : S \to 2^{\mathrm{AP}}$ — разметка состояний событиями

Переход вида (s_1, g, X, s_2) называется переходом из состояния s_1 в состояние s_2 с предусловием g и сбросом всех часов из X и будет изображаться так: $s_1 \xrightarrow{g,X} s_2$

Временны́е автоматы: синтаксис

Автомат $\mathcal{A}$ представляет собой размеченный конечный ориентированный граф:

- Вершины — это состояния автомата
- Ду́ги, помеченные предусловиями и множествами часов — это переходы автомата
- Остальные компоненты автомата — это метки вершин

В связи с этим к временны́м автоматам будет применяться графовая терминология

Временно́е ограничение $\mathbb{t}$ и множество часов $\emptyset$ в изображениях иногда будут опускаться, и множество часов $\{x_1, \ldots, x_n\}$ иногда будет записываться без фигурных скобок: $x_1, \ldots, x_n$

Временны́е автоматы: синтаксис

Замечание для любопытных

При разработке временного автомата может возникнуть желание записать ограничение «$x < k$» и «$x \leq k$», где $k \in \mathbb{R}_{\geq 0}$ или $k \in \mathbb{Q}_{\geq 0}$ (это множество всех неотрицательных рациональных чисел)

Принято полагать, что более строгое ограничение «$k \in \mathbb{N}_0$» на самом деле не ограничивает выразительные возможности:

- Любое число из $\mathbb{R}_{\geq 0}$ может быть приближено числом из $\mathbb{Q}_{\geq 0}$ с любой наперёд заданной точностью
- Замеры времени выполнения СРВ возможны только с некоторой погрешностью (точностью)
  - $\Rightarrow$ множество $\mathbb{R}_{\geq 0}$ излишне, достаточно использовать $\mathbb{Q}_{\geq 0}$
- Любой конечный набор рациональных чисел можно привести к общему знаменателю
- Домножением всех рациональных чисел в записи автомата на их общий знаменатель N («замедлением» модельного времени в N раз) можно преобразовать все эти числа в целые неотрицательные

Временны́е автоматы: семантика

Вычислительная конфигурация автомата $\mathcal{A}$ — это пара (s, ν), где s — состояние автомата и ν — оценка часов

Для технической простоты иногда будем полагать, что часы автомата упорядочены ($\mathcal{C} = \{x_1, \ldots, x_n\}$), и записывать оценку ν как набор значений часов: $(\nu(x_1), \ldots, \nu(x_n))$

Начальная конфигурация автомата $\mathcal{A}$ с начальным состоянием s_0 имеет вид $(s_0, (0, 0, \ldots, 0))$

Автомат $\mathcal{A}$ выполняется пошагово согласно двуместному отношению шага вычисления $\to_{\mathcal{A}}$

Это отношение зададим как объединение двух отношений, отвечающих двум возможностям шага вычисления автомата (строгое определение будет дальше):

- $\mapsto_{\mathcal{A}}$ — продвижение времени: время течёт, автомат бездействует
- $\hookrightarrow_{\mathcal{A}}$ — выполнение перехода: время не течёт, переход выполняется (мгновенно)

Иногда будем опускать индекс $\mathcal{A}$ в отношениях $\to_{\mathcal{A}}$, $\hookrightarrow_{\mathcal{A}}$ и $\mapsto_{\mathcal{A}}$, если автомат $\mathcal{A}$ однозначно задаётся контекстом или неважен

Временны́е автоматы: семантика

Продвижение времени

Для оценки часов ν, конфигураций $\sigma = (s, \nu)$ и σ' и числа́ $d \in \mathbb{R}_{\geq 0}$ будем использовать такие обозначения:

- $\nu + d$ — это оценка часов, такая что $(\nu + d)(x) = \nu(x) + d$ для любых часов x
- $\sigma + d = (s, \nu + d)$
- $\sigma \overset{d}{\mapsto} \sigma'$ означает, что $\sigma' = \sigma + d$

$\sigma \mapsto_{\mathcal{A}} \sigma'$ для автомата $\mathcal{A} = (S, s_0, \mathcal{I}, T, L)$ и конфигураций $\sigma = (s, \nu)$ и σ', если существует константа $d \in \mathbb{R}_{>0}$, для которой верно:

1. $\sigma \overset{d}{\mapsto} \sigma'$
2. $\nu + d \models \mathcal{I}(s)$

Временны́е автоматы: семантика

Выполнение перехода

Для оценки часов ν, множества часов X, конфигураций $\sigma = (s, \nu)$ и σ', состояния s' и перехода $t = (s \xrightarrow{g,X} s')$ будем использовать такие обозначения:

- $\nu[X]$ — оценка часов, такая что
$$\begin{aligned} \nu[X](x) &= 0, && \text{если } x \in X, \text{ и} \\ \nu[X](x) &= \nu(x) && \text{иначе} \end{aligned}$$
- $\sigma[X] = (s, \nu[X])$
- $\sigma[s'] = (s', \nu)$
- $\sigma \overset{t}{\hookrightarrow} \sigma'$ означает, что $\sigma' = \sigma[X][s']$

$\sigma \hookrightarrow_{\mathcal{A}} \sigma'$ для автомата $A = (S, s_0, \mathcal{I}, T, L)$ и конфигураций $\sigma = (s, \nu)$ и σ', если существует переход $t = (s \xrightarrow{g,X} s') \in T$, для которого верно:

1. $\nu \models g$
2. $\sigma \overset{t}{\hookrightarrow} \sigma'$
3. $\nu[X] \models \mathcal{I}(s')$

Временны́е автоматы: семантика

Трассой временно́го автомата $\mathcal{A}$ из конфигурации σ (или, коротко, — σ-трассой) назовём последовательность конфигураций вида

$$\sigma_1 \to_{\mathcal{A}} \sigma_2 \to_{\mathcal{A}} \sigma_3 \to_{\mathcal{A}} \dots,$$

где $\sigma_1 = \sigma$

σ-трассу автомата $\mathcal{A}$ назовём начальной, если σ — начальная конфигурация $\mathcal{A}$

Конфигурацию σ автомата $\mathcal{A}$ назовём тупиковой, если не существует конфигурации σ', такой что $\sigma \to_{\mathcal{A}} \sigma'$

Трассу назовём

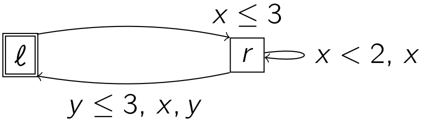
- тупиковой, если она конечна и оканчивается тупиковой конфигурацией, и
- полной, если она бесконечная или тупиковая

Вычислением автомата $\mathcal{A}$ назовём полную начальную трассу

Временны́е автоматы: семантика

Пример

Рассмотрим такой временной автомат $\mathcal{A}$ над множеством часов $\{x, y\}$ (атомарные высказывания опущены за ненадобностью):



Пример тупикового вычисления $\mathcal{A}$ *для порядка часов* (x, y):

$$(\ell, 0, 0) \hookrightarrow (r, 0, 0) \mapsto (r, 1, 1) \mapsto (r, \sqrt{2}, \sqrt{2}) \hookrightarrow (r, 0, \sqrt{2}) \mapsto (r, 3, \sqrt{2} + 3)$$

Пример бесконечного вычисления:

$$(\ell, 0, 0) \mapsto (\ell, 1, 1) \mapsto (\ell, 2, 2) \mapsto \ldots \mapsto (\ell, n, n) \mapsto \ldots$$

Другой пример бесконечного вычисления:

$$(\ell, 0, 0) \mapsto (\ell, 1.2, 1.2) \hookrightarrow (r, 1.2, 1.2) \hookrightarrow (\ell, 0, 0) \mapsto (\ell, 1.2, 1.2) \hookrightarrow \ldots$$