TVA : Techniques de Vérification Avancées

Nathalie Bertrand
nathalie.bertrand@inria.fr

Master 2 Recherche en Informatique

Automne 2013

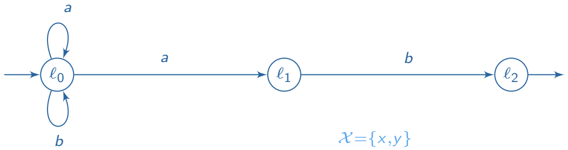
Part I

Timed automata

Outline

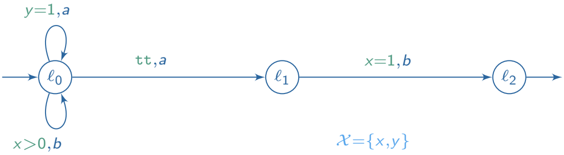
The model, informally

Timed automaton: Finite automaton enriched with clocks.



The model, informally

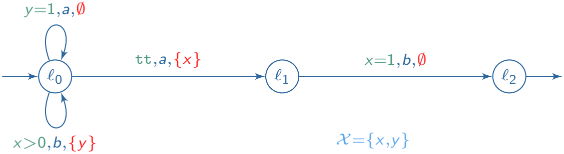
Timed automaton: Finite automaton enriched with clocks.



Transitions are equipped with guards

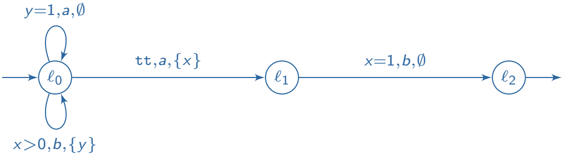
The model, informally

Timed automaton: Finite automaton enriched with clocks.



Transitions are equipped with guards and sets of reset clocks.

Syntax



Timed automata

A timed automaton is a tuple $\mathcal{A} = (L, L_0, L_{acc}, \Sigma, \mathcal{X}, E)$ with

- L finite set of locations $\qquad L = \{\ell_0, \ell_1, \ell_2\}$
- $L_0 \subseteq L$ initial locations $\qquad L_0 = \{\ell_0\}$
- $L_{acc} \subseteq L$ set of accepting locations $\qquad L_{acc} = \{\ell_2\}$
- Σ finite alphabet $\qquad \Sigma = \{a, b\}$
- $\mathcal{X}$ finite set of clocks $\qquad \mathcal{X} = \{x, y\}$
- $E \subseteq L \times \mathcal{G} \times \Sigma \times 2^{\mathcal{X}} \times L$ set of edges $\qquad \ell_0 \xrightarrow{x>0, a, \{y\}} \ell_0$
  where $\mathcal{G} = \{\bigwedge x \bowtie c \mid x \in \mathcal{X}, c \in \mathbb{N}\}$ is the set of guards.
  (with $\bowtie \in \{<, \leq, =, \geq, >\}$)

Outline

Semantics

Valuation: $v \in \mathbb{R}_+^{\mathcal{X}}$ assigns to each clock a clock-value

State: $(\ell, v) \in L \times \mathbb{R}_+^{\mathcal{X}}$ composed of a location and a valuation.

Transitions between states of $\mathcal{A}$:

- Delay transitions: $(\ell, v) \xrightarrow{\tau} (\ell, v + \tau)$
- Discrete transitions: $(\ell, v) \xrightarrow{a} (\ell', v')$

$$\text{if } \exists(\ell, g, a, Y, \ell') \in E \text{ with } v \models g \text{ and } \begin{cases} v'(x) = 0 & \text{if } x \in Y, \\ v'(x) = v(x) & \text{otherwise.} \end{cases}$$

Run of $\mathcal{A}$:
$(\ell_0, v_0) \xrightarrow{\tau_1} (\ell_0, v_0 + \tau_1) \xrightarrow{a_1} (\ell_1, v_1) \xrightarrow{\tau_2} (\ell_1, v_1 + \tau_2) \xrightarrow{a_2} \cdots \xrightarrow{a_k} (\ell_k, v_k)$
or simply: $(\ell_0, v_0) \xrightarrow{\tau_1, a_1} (\ell_1, v_1) \xrightarrow{\tau_2, a_2} \cdots \xrightarrow{\tau_k, a_k} (\ell_k, v_k)$

Semantics (cont.)

Time sequence: $\mathbf{t} = (t_i)_{1 \leq i \leq k}$ finite non-decreasing sequence over $\mathbb{R}_+$.

Timed word: $w = (\sigma, \mathbf{t}) = (a_i, t_i)_{1 \leq i \leq k}$ where $a_i \in \Sigma$ and $\mathbf{t}$ time sequence.

Accepted timed word

A timed word $w = (a_0, t_0)(a_1, t_1) \ldots (a_k, t_k)$ is accepted in $\mathcal{A}$,
if there is a run $\rho = (\ell_0, v_0) \xrightarrow{\tau_0, a_0} (\ell_1, v_1) \xrightarrow{\tau_1, a_1} \ldots (\ell_{k+1}, v_{k+1})$
with $\ell_0 \in L_0$, $\ell_{k+1} \in L_{acc}$, and $t_i = \sum_{j<i} \tau_j$.

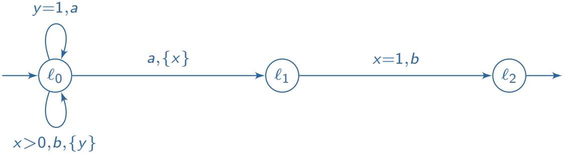
Accepted timed language: $\mathcal{L}(\mathcal{A}) = \{w \mid w \text{ accepted by } \mathcal{A}\}$.

Outline

Back to the example

NB: In the examples, we omit

- the guard when it is equivalent to tt, and
- the reset set when it is empty.

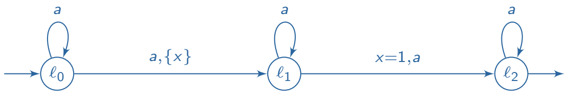


$w = (b, 0.1)(b, 0.3)(a, 1.3)(b, 1.5)(a, 1.5)(b, 2.5)$ is an accepted timed word

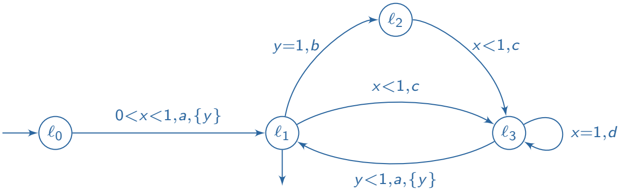
An accepting run for w is

$$(\ell_0, 0, 0) \xrightarrow{0.1,b} (\ell_0, 0.1, 0) \xrightarrow{0.2,b} (\ell_0, 0.3, 0) \xrightarrow{1,a} (\ell_0, 1.3, 1)$$
$$\xrightarrow{0.2,b} (\ell_0, 1.5, 0) \xrightarrow{0,a} (\ell_1, 0, 0) \xrightarrow{1,b} (\ell_2, 1, 1)$$

More examples



$$\mathcal{L}(\mathcal{A}) = \{(a, t_1)\cdots(a, t_k) | \exists i < j,\ t_j - t_i = 1\}$$



Does there exist an accepted timed word containing action b?

Outline

Region partitioning

Let $\mathcal{A}$ be a timed automaton with set of clocks $\mathcal{X}$ and set of constraints $\mathcal{C}$.
Let $\mathcal{R}$ be a finite partition of $\mathbb{R}_+^{\mathcal{X}}$, the set of valuations.

Set of regions

$\mathcal{R}$ is a set of regions (for $\mathcal{C}$) if

1. for every $g \in \mathcal{C}$ and for every $R \in \mathcal{R}$, $R \subseteq [\![g]\!]$ or $[\![g]\!] \cap R = \emptyset$,
2. for all $R, R' \in \mathcal{R}$, if there exists $v \in R$ and $t \in \mathbb{R}$ with $v + t \in R'$ then for every $v' \in R$ there exists $t' \in \mathbb{R}$ with $v' + t' \in R'$, and
3. for all $R, R' \in \mathcal{R}$, for every $Y \subseteq \mathcal{X}$ if $R_{[Y \leftarrow 0]} \cap R' \neq \emptyset$, then $R_{[Y \leftarrow 0]} \subseteq R'$.

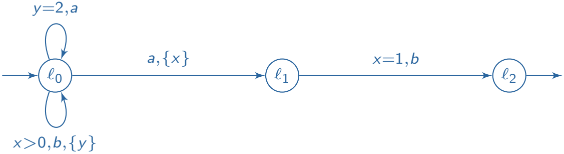
Let M be the maximal constant in $\mathcal{A}$.
The following equivalence relation yields the set of standard regions:

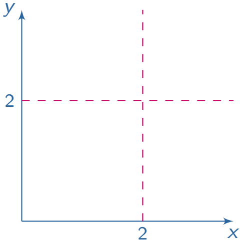
$$v \equiv^M v' \text{ if for every } x, y \in \mathcal{X}$$

- $v(x) > M \Leftrightarrow v'(x) > M$
- $v(x) \leq M \Rightarrow \Big((\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor) \text{ and } (\{v(x)\} = 0 \Leftrightarrow \{v'(x)\} = 0) \Big)$
- $(v(x) \leq M \text{ and } v(y) \leq M) \Rightarrow (\{v(x)\} \leq \{v(y)\} \Leftrightarrow \{v'(x)\} \leq \{v'(y)\})$

Regions with 2 clocks

Standard regions for 2 clocks can be represented in 2 dimensions.



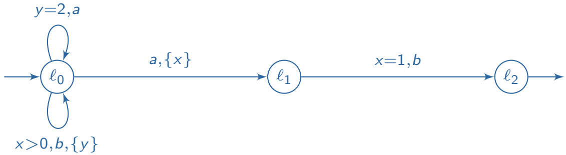


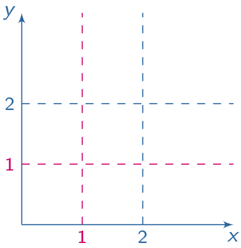
$v \equiv^M v'$ if for every $x, y \in \mathcal{X}$

- $v(x) > M \Leftrightarrow v'(x) > M$
- $v(x) \leq M \Rightarrow \Big((\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor)$ and $(\{v(x)\} = 0 \Leftrightarrow \{v'(x)\} = 0) \Big)$
- $(v(x) \leq M \text{ and } v(y) \leq M)$ $\Rightarrow (\{v(x)\} \leq \{v(y)\} \Leftrightarrow \{v'(x)\} \leq \{v'(y)\})$

Regions with 2 clocks

Standard regions for 2 clocks can be represented in 2 dimensions.



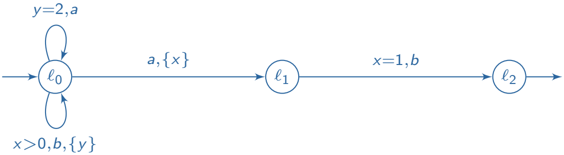


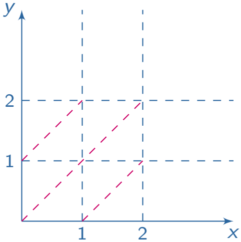
$v \equiv^M v'$ if for every $x, y \in \mathcal{X}$

- $v(x) > M \Leftrightarrow v'(x) > M$
- $v(x) \leq M \Rightarrow \Big((\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor)$ and $(\{v(x)\} = 0 \Leftrightarrow \{v'(x)\} = 0) \Big)$
- $(v(x) \leq M \text{ and } v(y) \leq M)$ $\Rightarrow (\{v(x)\} \leq \{v(y)\} \Leftrightarrow \{v'(x)\} \leq \{v'(y)\})$

Regions with 2 clocks

Standard regions for 2 clocks can be represented in 2 dimensions.



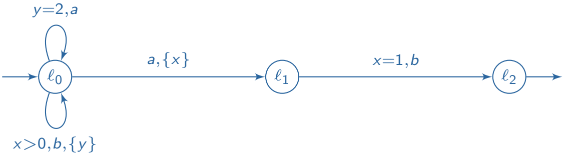


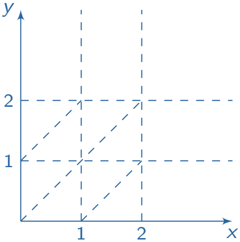
$v \equiv^M v'$ if for every $x, y \in \mathcal{X}$

- $v(x) > M \Leftrightarrow v'(x) > M$
- $v(x) \leq M \Rightarrow \Big((\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor)$ and $(\{v(x)\} = 0 \Leftrightarrow \{v'(x)\} = 0) \Big)$
- $(v(x) \leq M \text{ and } v(y) \leq M)$ $\Rightarrow (\{v(x)\} \leq \{v(y)\} \Leftrightarrow \{v'(x)\} \leq \{v'(y)\})$

Regions with 2 clocks

Standard regions for 2 clocks can be represented in 2 dimensions.





$v \equiv^M v'$ if for every $x, y \in \mathcal{X}$

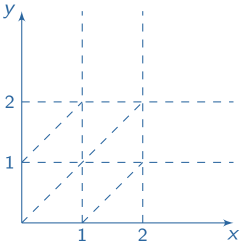
- $v(x) > M \Leftrightarrow v'(x) > M$
- $v(x) \leq M \Rightarrow \Big((\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor)$ and $(\{v(x)\} = 0 \Leftrightarrow \{v'(x)\} = 0) \Big)$
- $(v(x) \leq M \text{ and } v(y) \leq M)$ $\Rightarrow (\{v(x)\} \leq \{v(y)\} \Leftrightarrow \{v'(x)\} \leq \{v'(y)\})$

The partition is compatible with constraints, time elapsing and resets.

Operations on region

For two clocks, the (bounded) regions have the following shapes:

$R_{[Y\leftarrow 0]}$ denotes the region obtained from R by resetting clocks in $Y \subseteq \mathcal{X}$.
R' is a time-successor of R if there exists $v' \in R'$, $v \in R$, $t \in \mathbb{R}_+$ with $v' = v + t$.

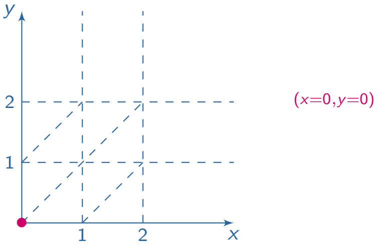


$(x{=}0, y{=}0)$

Operations on region

For two clocks, the (bounded) regions have the following shapes:

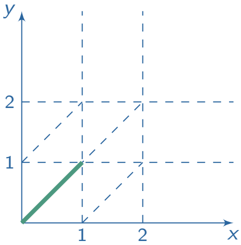
$R_{[Y\leftarrow 0]}$ denotes the region obtained from R by resetting clocks in $Y \subseteq \mathcal{X}$.
R' is a time-successor of R if there exists $v' \in R'$, $v \in R$, $t \in \mathbb{R}_+$ with $v' = v + t$.



Operations on region

For two clocks, the (bounded) regions have the following shapes:

$R_{[Y\leftarrow 0]}$ denotes the region obtained from R by resetting clocks in $Y \subseteq \mathcal{X}$.
R' is a time-successor of R if there exists $v' \in R'$, $v \in R$, $t \in \mathbb{R}_+$ with $v' = v + t$.

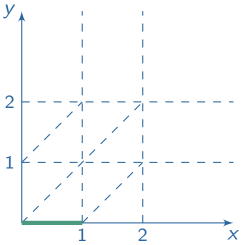


$(x{=}0, y{=}0) \xrightarrow{\text{delay}} (0{<}x{=}y{<}1)$

Operations on region

For two clocks, the (bounded) regions have the following shapes:

$R_{[Y \leftarrow 0]}$ denotes the region obtained from R by resetting clocks in $Y \subseteq \mathcal{X}$.
R' is a time-successor of R if there exists $v' \in R'$, $v \in R$, $t \in \mathbb{R}_+$ with $v' = v + t$.

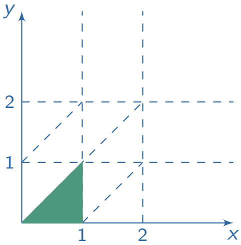


$(x{=}0, y{=}0) \xrightarrow{\text{delay}} (0{<}x{=}y{<}1) \xrightarrow{y:=0} (0{<}x{<}1, y{=}0)$

Operations on region

For two clocks, the (bounded) regions have the following shapes:

$R_{[Y \leftarrow 0]}$ denotes the region obtained from R by resetting clocks in $Y \subseteq \mathcal{X}$.
R' is a time-successor of R if there exists $v' \in R'$, $v \in R$, $t \in \mathbb{R}_+$ with $v' = v + t$.

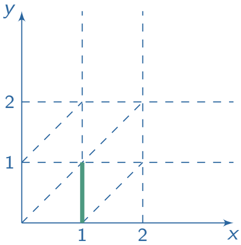


$(x{=}0, y{=}0) \xrightarrow{\text{delay}} (0{<}x{=}y{<}1) \xrightarrow{y:=0} (0{<}x{<}1, y{=}0) \xrightarrow{\text{delay}} (0{<}y{<}x{<}1)$

Operations on region

For two clocks, the (bounded) regions have the following shapes:

$R_{[Y\leftarrow 0]}$ denotes the region obtained from R by resetting clocks in $Y \subseteq \mathcal{X}$.
R' is a time-successor of R if there exists $v' \in R'$, $v \in R$, $t \in \mathbb{R}_+$ with $v' = v + t$.

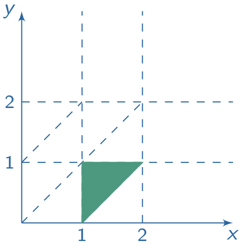


$(x{=}0, y{=}0) \xrightarrow{\text{delay}} (0{<}x{=}y{<}1) \xrightarrow{y:=0} (0{<}x{<}1, y{=}0) \xrightarrow{\text{delay}} (0{<}y{<}x{<}1) \xrightarrow{\text{delay}} (0{<}y{<}1{=}x)$

Operations on region

For two clocks, the (bounded) regions have the following shapes:

$R_{[Y\leftarrow 0]}$ denotes the region obtained from R by resetting clocks in $Y \subseteq \mathcal{X}$.
R' is a time-successor of R if there exists $v' \in R'$, $v \in R$, $t \in \mathbb{R}_+$ with $v' = v + t$.

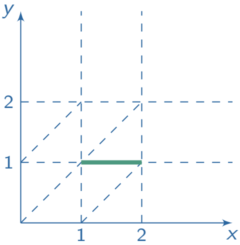


$(x{=}0,y{=}0) \xrightarrow{\text{delay}} (0{<}x{=}y{<}1) \xrightarrow{y:=0} (0{<}x{<}1,y{=}0) \xrightarrow{\text{delay}} (0{<}y{<}x{<}1) \xrightarrow{\text{delay}} (0{<}y{<}1{=}x) \xrightarrow{\text{delay}} (1{<}x{<}2, 0{<}y{<}1, \{x\}{<}\{y\})$

Operations on region

For two clocks, the (bounded) regions have the following shapes:

$R_{[Y \leftarrow 0]}$ denotes the region obtained from R by resetting clocks in $Y \subseteq \mathcal{X}$.
R' is a time-successor of R if there exists $v' \in R'$, $v \in R$, $t \in \mathbb{R}_+$ with $v' = v + t$.

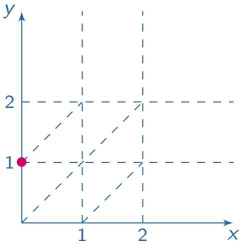


$(x{=}0, y{=}0) \xrightarrow{\text{delay}} (0{<}x{=}y{<}1) \xrightarrow{y:=0} (0{<}x{<}1, y{=}0) \xrightarrow{\text{delay}} (0{<}y{<}x{<}1) \xrightarrow{\text{delay}} (0{<}y{<}1{=}x) \xrightarrow{\text{delay}} (1{<}x{<}2, 0{<}y{<}1, \{x\}{<}\{y\}) \xrightarrow{\text{delay}} (y{=}1{<}x{<}2)$

Operations on region

For two clocks, the (bounded) regions have the following shapes:

$R_{[Y\leftarrow 0]}$ denotes the region obtained from R by resetting clocks in $Y \subseteq \mathcal{X}$.
R' is a time-successor of R if there exists $v' \in R'$, $v \in R$, $t \in \mathbb{R}_+$ with $v' = v + t$.



$(x{=}0,y{=}0) \xrightarrow{\text{delay}} (0{<}x{=}y{<}1) \xrightarrow{y:=0} (0{<}x{<}1,y{=}0) \xrightarrow{\text{delay}} (0{<}y{<}x{<}1) \xrightarrow{\text{delay}} (0{<}y{<}1{=}x) \xrightarrow{\text{delay}} (1{<}x{<}2,0{<}y{<}1,\{x\}{<}\{y\}) \xrightarrow{\text{delay}} (y{=}1{<}x{<}2) \xrightarrow{x:=0} (x{=}0,y{=}1)$

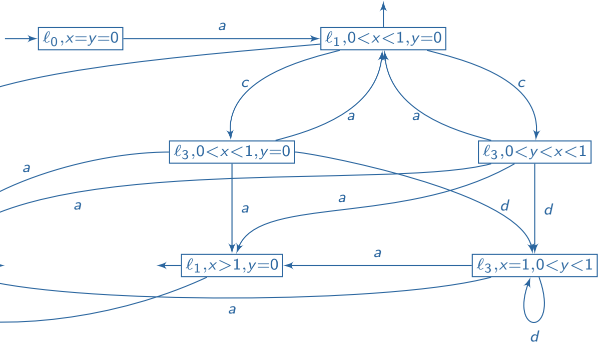
Outline

Region automaton: construction

From a timed automaton $\mathcal{A}$ we build a finite automaton $\alpha(\mathcal{A})$ as follows:

- States: $L \times \mathcal{R}$ Initial: $L_0 \times \mathcal{R}$ Final: $L_{acc} \times \mathcal{R}$
- Transitions:
  - $(\ell, R) \xrightarrow{a} (\ell', R')$ if there exists $\ell \xrightarrow{g,a,Y} \ell'$ in $\mathcal{A}$, there exists R'' time-successor of R with $R'' \subseteq [\![g]\!]$ and $R' = R''_{[Y \leftarrow 0]}$.

Example Region automaton for the second timed automaton of Slide 17.



Region automaton: properties

The number of states in $\alpha(\mathcal{A})$ is bounded by

$$|L| \cdot 2^{|\mathcal{X}|} \cdot |\mathcal{X}|! \cdot (2M+2)^{|\mathcal{X}|}$$

$\mathrm{Untime}(\mathcal{L}(\mathcal{A})) = \{\sigma | (\sigma, \mathbf{t}) \in \mathcal{L}(\mathcal{A})\} \subseteq \Sigma^*$ is the untimed language of $\mathcal{A}$.

Property

$\mathrm{Untime}(\mathcal{L}(\mathcal{A})) = \mathcal{L}(\alpha(\mathcal{A}))$

Consequence: the untimed language of $\mathcal{A}$ is regular.

Justification of the region automaton

Time-abstract bisimulation

Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be timed automata.
$\equiv \subseteq (L_1 \times \mathbb{R}_+^{\mathcal{X}_1}) \times (L_2 \times \mathbb{R}_+^{\mathcal{X}_2})$ is a time-abstract bisimulation between $\mathcal{A}_1$ and $\mathcal{A}_2$ if

- if $(\ell_1, v_1) \equiv (\ell_2, v_2)$ and $(\ell_1, v_1) \xrightarrow{\tau_1} (\ell_1, v_1 + \tau_1)$ for some $\tau_1 \in \mathbb{R}_+$, then there exists $\tau_2 \in \mathbb{R}_+$ with $(\ell_2, v_2) \xrightarrow{\tau_2} (\ell_2, v_2 + \tau_2)$ and $(\ell_1, v_1 + \tau_1) \equiv (\ell_2, v_2 + \tau_2)$
- if $(\ell_1, v_1) \equiv (\ell_2, v_2)$ and $(\ell_1, v_1) \xrightarrow{a} (\ell_1', v_1')$ for some $a \in \Sigma$, then there exists (ℓ_2', v_2') with $(\ell_2, v_2) \xrightarrow{a} (\ell_2', v_2')$ and $(\ell_1', v_1') \equiv (\ell_2', v_2')$
- and vice versa.

Let $\mathcal{A}$ be a timed automaton with maximal constant M.

Regions and time-abstract bisimulation

The relation $\equiv_M$ is a time-abstract bisimulation with finite index.

Outline

Reachability problem

Input: $\mathcal{A}$ timed automaton, ℓ location of $\mathcal{A}$
Question: is location ℓ reachable in $\mathcal{A}$?

Reachability problem

Reachability is decidable for timed automata. It is a PSPACE-complete problem.

Proof

- PSPACE-membership:
  - ℓ is reachable in $\mathcal{A}$ if and only if (ℓ, R) is reachable in $\alpha(\mathcal{A})$ for some R.
  - reachability is in NLOGSPACE for finite automata
  - $\alpha(\mathcal{A})$ has exponentially more states than $\mathcal{A}$
- PSPACE-hardness: reduction of the termination problem for a Turing machine with linearly bounded work space.

Outline

Universality and language inclusion

Universality

Input: $\mathcal{A}$ timed automaton
Question: does $\mathcal{A}$ accept all timed words?

Undecidability result

Universality is undecidable for timed automata.

Language inclusion

Input: $\mathcal{A}_1$, $\mathcal{A}_2$ timed automata
Question: $\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$?

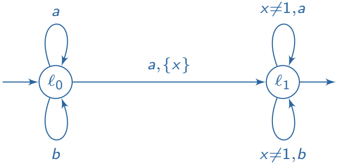
Corollary: Language inclusion in undecidable for timed automata.

Complementation

Non-closure

Timed automata are not closed under complement.

Proof hint The automaton below accepts a timed language whose complement cannot be recognized by a timed automaton.



Determinization

Deterministic TA

$\mathcal{A}$ is deterministic if $|L_0| = 1$ and for each $\ell \in L$, for every $a \in \Sigma$, $\ell \xrightarrow{g_1, a, Y_1} \ell_1$ and $\ell \xrightarrow{g_2, a, Y_2} \ell_2$ implies $[\![g_1]\!] \cap [\![g_2]\!] = \emptyset$.

If $\mathcal{A}$ is deterministic, there is at most one run on each timed word.

Closure

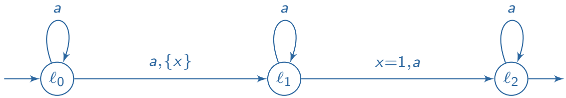
Deterministic timed automata are closed under complementation.

Expressivity

Timed automata are strictly more expressive than deterministic ones.

Determinizability

Example The automaton below accepts a timed language which cannot be recognized by a deterministic timed automaton.



Determinizability

Telling whether a timed automaton can be determinized is undecidable.

Outline

1. Introduction to timed automata
   - Model
   - Timed language
   - Examples
2. Region abstraction
   - Regions
   - Region automaton
   - Reachability problem
3. Limits of the finite abstraction
4. TCTL model checking
5. Algorithmics and implementation

Timed Computation Tree Logic

Real-time variant of CTL to express properties on timed automata where locations are labeled with sets of atomic propositions.

Syntax of TCTL

- state formulae: $\psi ::= \mathtt{tt} \mid a \mid g \mid \psi_1 \wedge \psi_2 \mid \neg\psi \mid \exists\varphi \mid \forall\varphi$
- path formulae: $\varphi ::= \psi_1 \mathsf{U}^J \psi_2$

$g \in \mathcal{G}$ is a guard and $J \subseteq \mathbb{R}_+$ is an interval with integer bounds.

Shorthands: $\Diamond^J\varphi \equiv \mathtt{tt}\mathsf{U}^J\varphi$, $\exists\Box^J\varphi \equiv \neg\forall\Diamond^J\neg\varphi$, $\forall\Box^J\varphi \equiv \neg\exists\Diamond^J\neg\varphi$.

Examples:

- $\mathtt{no\_error}\ \mathsf{U}^{[0,30]}\ \mathtt{deadlock}$
- $\forall\Box(\mathtt{on} \implies \forall\Diamond^{\leq 2}\neg\mathtt{on})$
- $\forall\Box\Big((\mathtt{near} \wedge (y=0)) \implies \forall\Box^{\leq 2}(\neg\mathtt{in})\Big)$

Satisfaction of TCTL formulas

TCTL formulae are interpreted over time-divergent runs only!

Time-divergence

The infinite run $(\ell_0, v_0) \xrightarrow{\tau_1, a_1} (\ell_1, v_1) \xrightarrow{\tau_2, a_2} \cdots$ is time-divergent if $\sum_k \tau_k = \infty$.

Semantics of state formulae

- $(\ell, v) \models g$ if and only if $v \models g$
- $(\ell, v) \models \exists\varphi$ if and only if there exists a time-divergent run ρ with $\rho \models \varphi$

Semantics of path formulae (Until modality)
for time-divergent run $\rho = (\ell_0, v_0) \xrightarrow{\tau_1, a_1} (\ell_1, v_1) \xrightarrow{\tau_2, a_2} \cdots$
$\rho \models \psi_1 \mathsf{U}^J \psi_2$ if and only if there exists $i \geq 0$, there exists $\tau \in [0, \tau_i]$ such that

- $(\ell_i, v_i + \tau) \models \psi_2$ with $\sum_{k=1}^{i} \tau_k + \tau \in J$,
- $\forall j \leq i$, $\forall \tau' \in [0, \tau_j]$,
  $\sum_{k=1}^{j} \tau_k + \tau' \leq \sum_{k=1}^{i} \tau_k + \tau \implies (\ell_j, v_j + \tau') \models \psi_1 \vee \psi_2$

TCTL model checking: overview

From a timed automaton $\mathcal{A}$ and a TCTL formula ψ build:

- $\alpha(\mathcal{A}, \psi)$ a region automaton taking ψ into account
- $\hat{\psi}$ a CTL formula

such that $\mathcal{A} \models_{TCTL} \psi \iff \alpha(\mathcal{A}, \psi) \models_{CTL} \hat{\psi}$

TCTL: elimination of timing parameters

For valuation $v \in \mathbb{R}_+^{\mathcal{X}}$, and additional clock $z \notin \mathcal{X}$

$v\{z := t\} \in \mathbb{R}_+^{\mathcal{X} \cup \{z\}}$ such that $\begin{cases} v\{z := t\}(z) = t \\ v\{z := t\}(x) = v(x) & \text{for } x \in \mathcal{X} \end{cases}$

$\mathcal{A}$ timed automaton over clocks $\mathcal{X}$, z additional clock.

Elimination of timings in $\psi_1 \mathsf{U}^J \psi_2$

- $s \models \exists\big(\psi_1 \mathsf{U}^J \psi_2\big) \iff s\{z := 0\} \models \exists\Big(\big(\psi_1 \vee \psi_2\big)\mathsf{U}\big((z \in J) \wedge \psi_2\big)\Big)$
- $s \models \forall\big(\psi_1 \mathsf{U}^J \psi_2\big) \iff s\{z := 0\} \models \forall\Big(\big(\psi_1 \vee \psi_2\big)\mathsf{U}\big((z \in J) \wedge \psi_2\big)\Big)$

Examples:

- $\exists\Diamond^{\geq 3}\psi \;\equiv\; \exists\Diamond\big((z \geq 3) \wedge \psi\big)$
- $\exists\Box^{\leq 2}\psi \;\equiv\; \exists\big((z \leq 2) \implies \psi\big)$

Model checking TCTL

- $Sat(\psi) = \{s \mid s \models \psi\}$ computed recursively by structural induction.

$$\psi = \exists\Box^{\leq 2} a \wedge \forall(b \mathsf{U}^{[2,3]} \exists(\Diamond^{\geq 3} c))$$

$$\begin{tikzcd}
& \wedge \arrow[dl] \arrow[dr] & & \\
\exists\Box^{\leq 2}(_) \arrow[d] & & \forall(_\mathsf{U}^{[2,3]}_) \arrow[dl] \arrow[dr] & \\
a & b & & \exists(\Diamond^{\geq 3}_) \arrow[d] \\
& & & c
\end{tikzcd}$$

- $Sat\big(\exists(S_1 \mathsf{U}^J S_2)\big) = \{s \mid s\{z := 0\} \models \exists\Big(\big(S_1 \vee S_2\big)\mathsf{U}\big((z \in J) \wedge S_2\big)\Big)\}$

Complexity

Model checking of TCTL for timed automata is PSPACE-complete.

Outline

Symbolic model checking

Two general methods to solve the reachability problem.

Forward analysis

iterative computation
of successors of Init

Symbolic model checking

Two general methods to solve the reachability problem.

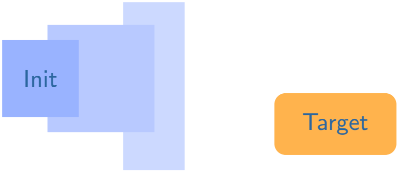
Forward analysis

iterative computation
of successors of Init

Symbolic model checking

Two general methods to solve the reachability problem.

Forward analysis

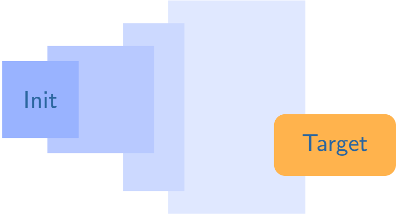


iterative computation
of successors of Init

Symbolic model checking

Two general methods to solve the reachability problem.

Forward analysis



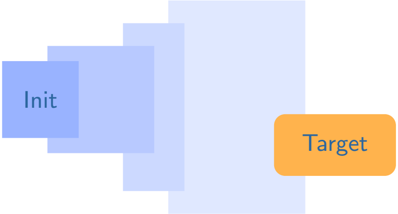
iterative computation
of successors of Init

Symbolic model checking

Two general methods to solve the reachability problem.

Forward analysis

Backward analysis



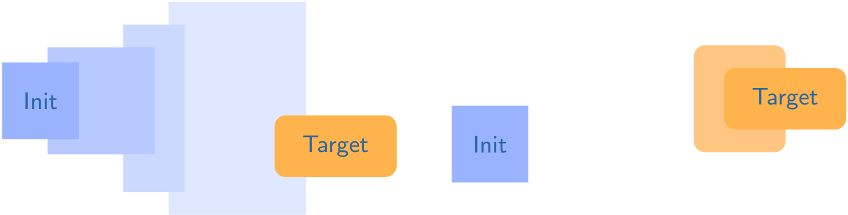
iterative computation of successors of Init

iterative computation of predecessors of Target

Symbolic model checking

Two general methods to solve the reachability problem.

Forward analysis **Backward analysis**



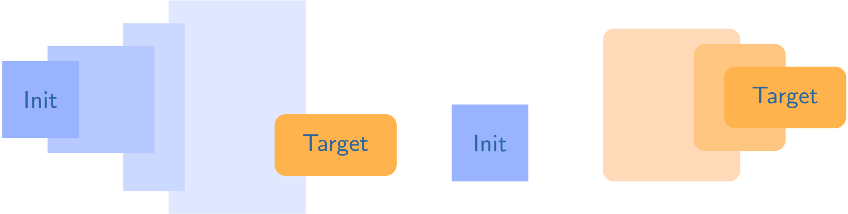
iterative computation of successors of Init

iterative computation of predecessors of Target

Symbolic model checking

Two general methods to solve the reachability problem.

Forward analysis | **Backward analysis**



iterative computation of successors of Init

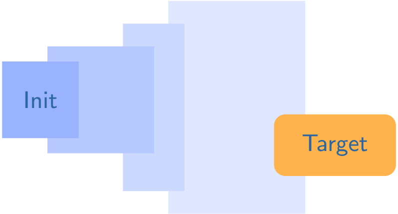
iterative computation of predecessors of Target

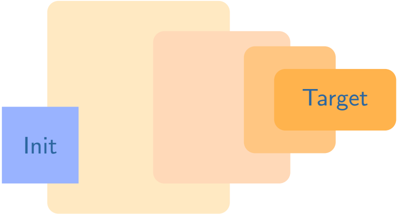
Symbolic model checking

Two general methods to solve the reachability problem.

Forward analysis

Backward analysis





iterative computation of successors of Init

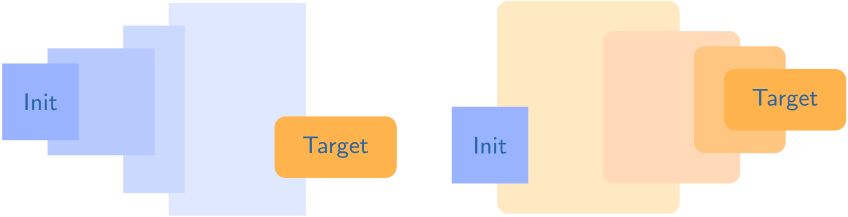
iterative computation of predecessors of Target

Symbolic model checking

Two general methods to solve the reachability problem.

Forward analysis

Backward analysis



iterative computation
of successors of Init

iterative computation
of predecessors of Target

Issues: Representation of the sets of states + Termination of the computation.

Zones

Zones are symbolic representations of sets of valuations.
A clock constraint g defines a zone $[\![g]\!] = \{v \in \mathbb{R}_+^{\mathcal{X}} | v \models g\}$.

For verification purposes, the following operations on zones Z, Z' are needed.

- forward analysis:
  - Future of Z: $\overrightarrow{Z} = \{v + t | v \in Z,\ t \in \mathbb{R}_+\}$
  - Reset in Z of clocks in $Y \subseteq \mathcal{X}$: $Z_{[Y \leftarrow 0]} = \{v_{[Y \leftarrow 0]} | v \in Z\}$
  - Intersection of Z and Z': $Z \cap Z' = \{v | v \in Z \text{ and } v \in Z'\}$
  - Emptiness test: decide if Z is empty.
- backward analysis:
  - Past of Z: $\overleftarrow{Z} = \{v - t | v \in Z,\ t \in \mathbb{R}_+\}$
  - Inverse reset: $Z_{[Y \leftarrow 0]^{-1}}$ the largest Z' with $Z'_{[Y \leftarrow 0]} = Z$
  - Intersection
  - Emptiness test

Data structure

Zones are represented by Difference Bounded Matrices (DBM).

Difference Bounded Matrix

A DBM over the set of n clocks $\mathcal{X}$ is an $(n+1)$-square matrix of pairs

$$(m, \prec) \text{ with } \prec \in \{<, \leq\} \text{ and } m \in \mathbb{Z} \cup \{\infty\}$$

$(m_{i,j}, \prec_{i,j})$ encodes the constraint $x_i - x_j \prec_{i,j} m_{i,j}$ (with convention $x_0 = 0$)

Example A DBM and the zone it represents.

$$\begin{array}{c} \\ 0 \\ x \\ y \end{array} \begin{array}{c} \begin{array}{ccc} 0 & x & y \end{array} \\ \begin{pmatrix} (\infty, <) & (-3, \leq) & (\infty, <) \\ (\infty, <) & (\infty, <) & (4, <) \\ (5, \leq) & (\infty, <) & (\infty, <) \end{pmatrix} \end{array}$$

$$x \geq 3 \wedge y \leq 5 \wedge x - y < 4$$

Normal form (via Floyd algorithm)

$$\begin{array}{c} \\ 0 \\ x \\ y \end{array} \begin{array}{c} \begin{array}{ccc} 0 & x & y \end{array} \\ \begin{pmatrix} (0, \leq) & (-3, \leq) & (0, \leq) \\ (9, <) & (0, \leq) & (4, <) \\ (5, \leq) & (2, \leq) & (0, \leq) \end{pmatrix} \end{array}$$

Comparison

Backward analysis

The backward analysis terminates and is correct.

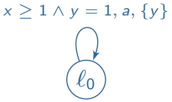
Proof Termination is based on the fact that finite union of regions are stable under the following operations: past $\overleftarrow{Z}$, inverse reset $Z_{[Y\leftarrow 0]^{-1}}$, and intersection $g \cap Z$.

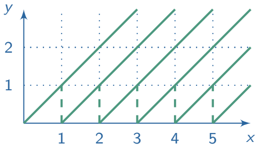
Forward analysis

The forward analysis is correct when it terminates.

Note that it may not terminate.

Example





Uppaal in a nutshell

Uppaal

- developed at Uppsala and Aalborg universities
- performs forward analysis (with extrapolation) for timed automata

http://www.uppaal.com/

References

- Seminal Paper

  - R. Alur and D. Dill. A Theory of Timed Automata. Theoretical Computer Science 126(2): 183-235. 1994.

- To go further...

  - O. Finkel. Undecidable Problems about Timed Automata. Proceedings of FORMATS'06, pages 187-199. 2006.
  - L. Aceto and F. Laroussinie. Is your model-checker on time? On the complexity of model checking for timed modal logics. Journal Logic Algebraic Programming 52-53: 7-51. 2002.
  - R. Alur and P. Madhusudan. Decision Problems for Timed Automata: A Survey. Proceedings of SFM-04, pages 1-24. 2004.
  - R. Alur, C. Courcoubetis, D. Dill. Model checking in dense real-time. Information and Computation 104(2): 2-34. 1993.

Part II

Probabilistic model checking

Probabilistic model checking

Models

- Discrete-time Markov chains (DTMC)
- Markov decision processes (MDP)
- Probabilistic timed automata (PTA)
- Continuous-time Markov chains (CTMC)

Logics

- Probabilistic Linear Temporal Logic (PLTL)
- Probabilistic Computation Tree Logic (PCTL)
- Probabilistic Timed Computation Tree Logic (PTCTL)
- Continuous Stochastic Logic (CSL)

Questions

- Qualitative: how does the probability compare to 0 and 1?
- Quantitative: compute/approximate the probability

Outline

6. Discrete-time Markov chains
   - Reachability properties
   - Branching-time model checking
   - Linear-time model checking

7. Markov decision processes
   - Reachability properties
   - Branching-time model checking
   - Linear-time model checking

8. Probabilistic timed automata

9. Continuous-time Markov chains

Outline

Motivation example

Tennis: Statistics give probability that Nadal wins a point when serving against Djokovic on clay.

- What is the probability that Nadal wins 4 points in a raw?
- What is the probability that Djokovic wins in 3 sets?

Discrete-time Markov chains

Discrete time Markov chain (DTMC)

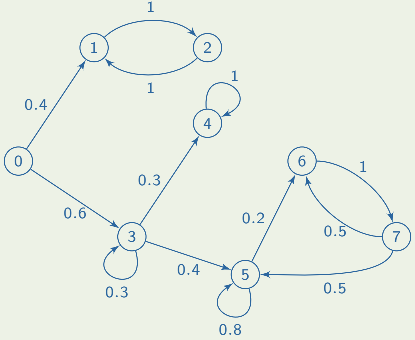
$\mathcal{M} = (S, \mathbf{P}, p_{\text{init}}, \text{lab}, AP)$ with

- S finite set of states, $\mathbf{P}$ probability matrix, p_{init} initial distribution, $\text{lab} : S \to 2^{AP}$ labels states with atomic propositions.

Example

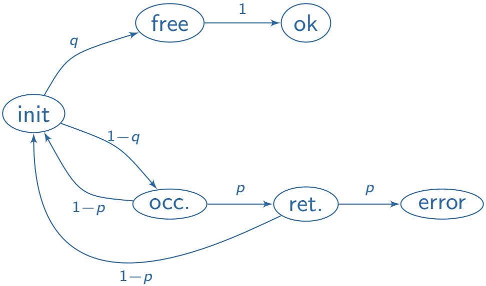
$S = \{0, 1, \cdots, 7\}$ $\quad p_{\text{init}} = (1, 0, \cdots, 0)$

$$\mathbf{P} = \begin{pmatrix} 0 & .4 & 0 & .6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .3 & .3 & .4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & .8 & .2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & .5 & .5 & 0 \end{pmatrix}$$



Example: Zeroconf protocol

IP address automatic allocation



- with high probability (q), a free IP address is randomly chosen;
- otherwise, the host with the same address sends an alert, which can be lost (with probability p);
- the host sends n probes (here $n = 2$) to increase the reliability.

Measure on DTMC paths

Probability of a finite path $\pi = s_0 s_1 \cdots s_n$:

$$Pr(\pi) = p_{init}(s_0) \prod_{i=0..n-1} \mathbf{P}_{i,i+1}.$$

Cylinder $\mathrm{Cyl}(\pi) = \{\pi_{max} | \pi \text{ prefix of } \pi_{max}\}$.

Probability measure

Pr is the unique probability measure on the σ-algebra generated by all $\mathrm{Cyl}(\pi)$, such that $Pr(\mathrm{Cyl}(\pi)) = Pr(\pi)$.

Outline

6 Discrete-time Markov chains
- Reachability properties
- Branching-time model checking
- Linear-time model checking

7 Markov decision processes
- Reachability properties
- Branching-time model checking
- Linear-time model checking

8 Probabilistic timed automata

9 Continuous-time Markov chains

Reachability properties

Goal: Compute $Pr(s_0 \models \Diamond T)$, for T set of target states.

For state $s \in S$, let $x_s = Pr(s \models \Diamond T)$.

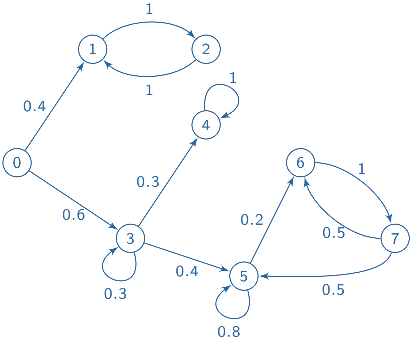
- $x_s = 1$ if $s \in T$
- $x_s = 0$ if $s \not\models E\Diamond T$
- $x_s = \sum_{t \in S} \mathbf{P}(s,t)\ x_t$

  $\longrightarrow$ resolution of a system of linear equations

Constrained reachability $Pr(s_0 \models T_1 \mathrm{U}\, T_2)$

- $x'_s = 1$ if $s \in T_2$
- $x'_s = 0$ if $s \notin T_1$ or $s \not\models E\Diamond T_2$
- $x'_s = \sum_{t \in S} \mathbf{P}(s,t)\ x'_t$

Example of probability computation



Computation of $Pr(0 \models \Diamond 6)$

$$\begin{cases} x_6 = 1 \\ x_1 = x_2 = x_4 = 0 \\ x_0 = 0.6\ x_3 \\ x_3 = 0.3\ x_3 + 0.4\ x_5 \\ x_5 = 0.8\ x_5 + 0.2 \\ x_7 = 0.5\ x_5 + 0.5 \end{cases}$$

$x_0 = \frac{12}{35}$

Outline

6 Discrete-time Markov chains
- Reachability properties
- Branching-time model checking
- Linear-time model checking

7 Markov decision processes
- Reachability properties
- Branching-time model checking
- Linear-time model checking

8 Probabilistic timed automata

9 Continuous-time Markov chains

Probabilistic Computation Tree Logic

Syntax of PCTL

- state formulae: $\psi ::= \mathtt{tt} \mid a \mid \psi_1 \wedge \psi_2 \mid \neg\psi \mid \mathbb{P}_J(\varphi)$
- path formulae: $\varphi ::= \bigcirc\psi \mid \psi_1 \mathsf{U} \psi_2 \mid \psi_1 \mathsf{U}_{\leq n} \psi_2$

$$s \models \mathbb{P}_J(\varphi) \quad \text{iff} \quad Pr(s \models \varphi) \in J$$

Shorthands: $\mathbb{P}_{[r,r]}(\varphi) \equiv \mathbb{P}_{=r}(\varphi)$, $\mathbb{P}_{(r,1]}(\varphi) \equiv \mathbb{P}_{>r}(\varphi)$, $\mathbb{P}_{[0,r]}(\varphi) \equiv \mathbb{P}_{\leq r}(\varphi)$, etc.

Measurability of PCTL events

For φ PCTL path formula and $\mathcal{M}$ a Markov chain, $\{\pi \mid \pi \models \varphi\}$ is measurable (so $Pr(s \models \varphi)$ is well-defined).

For ψ a PCTL state formula: $Sat(\psi) = \{s \mid s \models \psi\}$.

Comparison of PCTL and CTL

Qualitative PCTL

Fragment of PCTL where $J \in \{(0,1],[0,0],[1,1],[0,1)\}$.
In words, probabilities are only compared to 0 and 1.

Qualitative PCTL versus CTL

- $\forall\varphi \not\Longleftrightarrow \mathbb{P}_{=1}(\varphi)$
- In infinite Markov chains, there is no CTL formula equivalent to $\mathbb{P}_{=1}(\Diamond a)$.
- There is no qualitative PCTL formula equivalent to $\forall\Diamond a$.

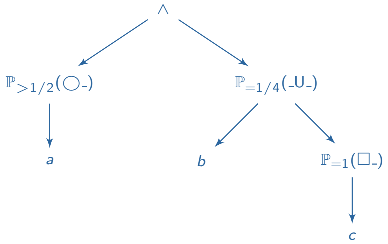
Qualitative PCTL and CTL are expressively incomparable.

For finite Markov chains, qualitative PCTL is similar to CTL with strong fairness.

Model checking PCTL

- $Sat(\psi) = \{s \mid s \models \psi\}$ computed recursively by structural induction.

$\psi = \mathbb{P}_{>1/2}(\bigcirc a) \wedge \mathbb{P}_{=1/4}(b \mathsf{U} \mathbb{P}_{=1}(\Box c))$



- $Sat\big(\mathbb{P}_J(S_1 \mathsf{U} S_2)\big) = \{s \mid Pr(s \models S_1 \mathsf{U} S_2) \in J\}$ constrained reachability.

Complexity

Model checking of PCTL for Markov chains is linear in the size of φ and polynomial in the size of $\mathcal{M}$.

Probabilistic bisimulation

Bisimulation for Markov chains

A probabilistic bisimulation for a Markov chain $\mathcal{M} = (S, \mathbf{P}, p_{\text{init}}, \text{lab}, AP)$ is an equivalence relation $\mathcal{R}$ on S such that for all states $(s_1, s_2) \in \mathcal{R}$

- $\text{lab}(s_1) = \text{lab}(s_2)$, and
- for each equivalence class $T \in S/\mathcal{R}$, $\sum_{t \in T} \mathbf{P}(s_1, t) = \sum_{t \in T} \mathbf{P}(s_2, t)$.

Probabilistic bisimulation and PCTL equivalence

Let $\mathcal{M}$ be a Markov chain, s_1, s_2 states of $\mathcal{M}$. Then

$$s_1 \text{ and } s_2 \text{ are bisimilar} \iff s_1 \text{ and } s_2 \text{ satisfy the same PCTL properties}$$

Outline

Linear temporal logic

Syntax of Linear Temporal Logic (LTL)

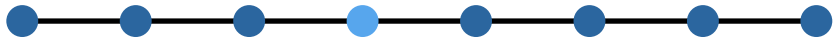
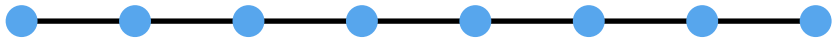
$$\varphi ::= \mathtt{tt} \mid a \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \bigcirc\, \varphi \mid \varphi_1 \mathsf{U} \varphi_2$$

where a is an atomic proposition.

$\bigcirc$ is the next step operator and U is the until operator.

Macros: eventually $\Diamond\varphi = \mathtt{tt}\mathsf{U}\varphi$, and always $\Box\varphi = \neg\Diamond\neg\varphi$ operators.

Intuitive semantics:

- $\bigcirc\varphi$ 
- $\varphi \mathsf{U} \varphi'$ 
- $\Diamond\varphi$ 
- $\Box\varphi$ 

Reminder: LTL model checking of transition systems

Given φ an LTL formula over AP, $[\![\varphi]\!] = \{\sigma \in (2^{AP})^\omega \mid \sigma \models \varphi\}$.

From LTL to automata

For every LTL formula φ, there exists a nondeterministic Büchi automaton (NBA) $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A}) = [\![\varphi]\!]$.

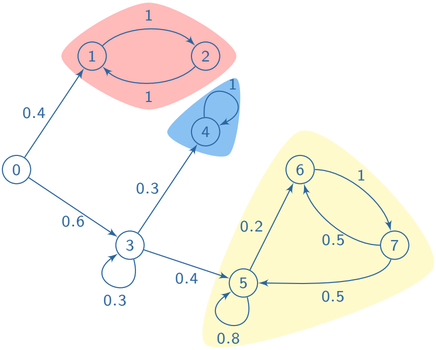
Online tool: LTL2BA

Automata-based LTL model checking

Input: transition system T and LTL formula φ over AP
Question: $T \models \varphi$?

- Build an NBA $\mathcal{A}_{\neg\varphi}$ such that $\mathcal{L}(\mathcal{A}_{\neg\varphi}) = [\![\neg\varphi]\!]$.
- Build the product transition system $P = T \otimes \mathcal{A}_{\neg\varphi}$
  - if there is a path in P satisfying the acceptance condition of $\mathcal{A}_{\neg\varphi}$, then return "no",
  - else return "yes".

Bottom Strongly Connected Components



Properties of BSCC

Let $\mathcal{C}$ be the set of basic strongly connected components in $\mathcal{M}$.

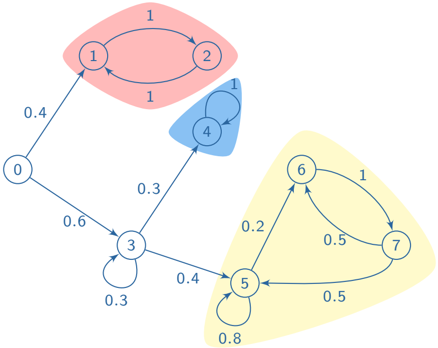
- $Pr(s_0 \models \Diamond \bigcup_{C \in \mathcal{C}} C) = 1$,
- $\forall s \in C (\in \mathcal{C}),\ Pr(s \models \bigwedge_{t \in C} \Box \Diamond t) = 1$.

Prefix-independent properties

Prefix-independency

A property is said prefix-independent if its validity only depends on the set of states that are visited infinitely often along a path.

For φ prefix-independent, $Pr(s_0 \models \varphi) = Pr(s_0 \models \Diamond\{C \in \mathcal{C} | C \models \varphi\})$.



Computation of $Pr(0 \models \Box\Diamond \text{ odd})$

$Pr(0 \models \Box\Diamond \text{ odd}) =$

$Pr(0 \models \Diamond C_{1,2}) + Pr(0 \models \Diamond C_{5,6,7})$

$Pr(0 \models \Box\Diamond \text{ odd}) = \frac{26}{35}$

Omega-regular properties

- Omega-regular property represented by Deterministic Rabin Automaton (DRA).
- Given $\mathcal{M}$ Markov chain and $\mathcal{A}$ DRA, what is the probability in $\mathcal{M}$ to generate traces in $\mathcal{L}(\mathcal{A})$?

- Rabin acceptance condition: $\bigvee_{1 \leq i \leq k} (\Diamond\Box\neg L_i \wedge \Box\Diamond K_i)$.
- A BSCC C in $\mathcal{M} \otimes \mathcal{A}$ is accepting if for some index i, $T \cap (S \times L_i) = \emptyset$ and $T \cap (S \times K_i) \neq \emptyset$.

Model checking omega-regular properties

Given $\mathcal{M}$ a DTMC, $\mathcal{A}$ a DRA.
Let U be the union of all accepting BSCCs in $\mathcal{M} \otimes \mathcal{A}$. Then:

$$Pr^{\mathcal{M}}(s_0 \models \mathcal{A}) = Pr^{\mathcal{M}\otimes\mathcal{A}}((s_0, q_0) \models \Diamond U).$$

Outline

Motivation example

Soccer: US goalkeeper Solo watched the last 30 penalty kicks by Necib.

- Should she move left or right to maximise the probability to stop the ball?

Markov decision processes

Markov decision process (MDP)

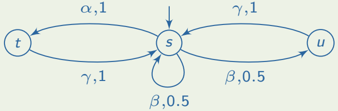
$\mathcal{M} = (S, Act, \mathbf{P}, p_{\mathrm{init}}, \mathrm{lab}, AP)$ with

- S finite set of states, p_{init} initial distribution, Act set of actions,
- $\mathbf{P} : S \times Act \times S \to [0,1]$ transition probability function s.t.

$$\forall s \in S,\ \forall \alpha \in Act,\ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \in \{0,1\},$$

- lab labels states with atomic propositions.

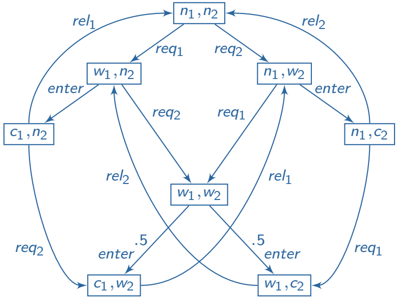
Example



nondeterministic choice in s between α and β.

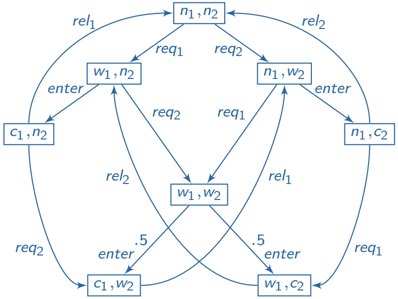
Example: mutual exclusion protocol

- two concurrent processes
- access to the critical section delivered by randomized arbiter



Example: mutual exclusion protocol

- two concurrent processes
- access to the critical section delivered by randomized arbiter

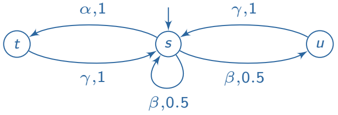


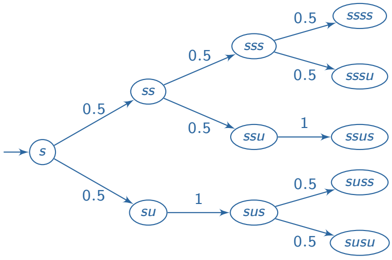
What is the probability that process 2 enters its critical section within 3 steps, assuming it is waiting?

Strategies

Strategy

Let $\mathcal{M} = (S, Act, \mathbf{P}, p_{\text{init}}, \text{lab}, AP)$ be an MDP. A strategy for $\mathcal{M}$ is a function $\sigma : S^+ \to Act$ s.t. the action $\sigma(s_0 s_1 \cdots s_n)$ is enabled in s_n.





Example of strategy σ:

- $\sigma(s) = \sigma(*ss) = \beta$
- $\sigma(*ts) = \beta$, $\sigma(*us) = \alpha$
- $\sigma(*t) = \sigma(*u) = \gamma$

adapter

MDP $\mathcal{M}$ + strategy σ = Markov chain $\mathcal{M}_\sigma$
Pr^σ probability measure in $\mathcal{M}_\sigma$

Outline

Reachability properties

Goal: Compute $Pr^{\max}(s \models \Diamond T) = \sup_\sigma Pr^\sigma(s \models \Diamond T)$.

For state $s \in S$, let $x_s = Pr^{\max}(s \models \Diamond T)$.

- $x_s = 1$ if $s \in T$
- $x_s = 0$ if $s \not\models E\Diamond T$
- $x_s = \max_{\alpha \in Act} \sum_{t \in S} \mathbf{P}(s, \alpha, t)\ x_t$

  $\longrightarrow$ resolution of a linear program

Constrained reachability $Pr^{\max}(s \models T_1 \mathsf{U} T_2)$

- $x'_s = 1$ if $s \in T_2$
- $x'_s = 0$ if $s \notin T_1$ or $s \not\models E\Diamond T_2$
- $x'_s = \max_{\alpha \in Act} \sum_{t \in S} \mathbf{P}(s, \alpha, t)\ x'_t$

Memoryless strategies

For reachability and constrained reachability properties, there exists a memoryless strategy $\sigma : S \to Act$ that maximizes the probability.

Outline

6. Discrete-time Markov chains
   - Reachability properties
   - Branching-time model checking
   - Linear-time model checking

7. Markov decision processes
   - Reachability properties
   - Branching-time model checking
   - Linear-time model checking

8. Probabilistic timed automata

9. Continuous-time Markov chains

Model checking PCTL

Probabilistic operator $\mathbb{P}_J(.)$ bounds the probability for all strategies.

$$s \models \mathbb{P}_J(\varphi) \iff \forall\sigma,\ Pr^{\sigma}(s \models \varphi) \in J$$
$$\iff [Pr^{\min}(s \models \varphi), Pr^{\max}(s \models \varphi)] \subseteq J$$

Recursive computation of $Sat(\psi)$ by structural induction on ψ.

- $\psi = \mathbb{P}_J(\psi_1 \mathsf{U} \psi_2)$
  - $p_s^+ = Pr^{\max}(s \models \psi_1 \mathsf{U} \psi_2)$
  - $p_s^- = Pr^{\min}(s \models \psi_1 \mathsf{U} \psi_2)$
  - $Sat(\psi) = \{s \mid [p_s^-, p_s^+] \subseteq J\}$
- $\psi = \mathbb{P}_J(\bigcirc \psi')$
  - $p_s^+ = Pr^{\max}(s \models \bigcirc \psi') = \max_{\alpha \in Act} \sum_{t \in Sat(\psi')} \mathbf{P}(s, \alpha, t);$
  - $p_s^- = Pr^{\min}(s \models \bigcirc \psi') = \min_{\alpha \in Act} \sum_{t \in Sat(\psi')} \mathbf{P}(s, \alpha, t);$
  - $Sat(\psi) = \{s \mid [p_s^-, p_s^+] \subseteq J\}.$

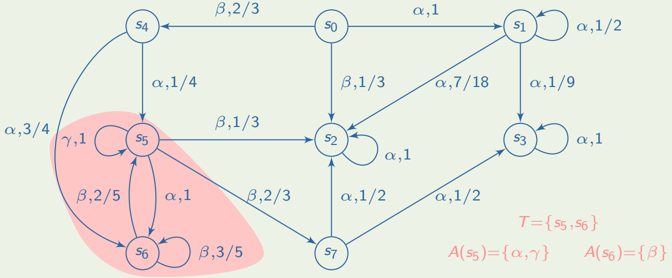
Outline

End components

End component

A sub-MDP (T, A) of $\mathcal{M}$ is an end component if it is strongly connected.

Example



Recurrence property

For each end component (T, A) there exists a strategy σ that ensures for every $s \in T$, $Pr^{\sigma}(s \models \Box T \wedge \bigwedge_{t \in T} \Box\Diamond t) = 1$.

Prefix-independent properties

Limit behaviors of MDP

Under each strategy, for almost every path, the states visited infinitely often and the actions taken infinitely often form an end component.

To a prefix-independent property φ we associate

- U_φ the union of the sets T such that (T, A) is an end component with $T \models \varphi$, and
- V_φ the union of the sets T such that (T, A) is an end component with $\neg(T \models \varphi)$.

Verifying prefix-independent properties

- $Pr^{\max}(s \models \varphi) = Pr^{\max}(s \models \Diamond U_\varphi)$
- $Pr^{\min}(s \models \varphi) = 1 - Pr^{\max}(s \models \Diamond V_\varphi)$
- Finite-memory strategies are sufficient for extremal probabilities.

Outline

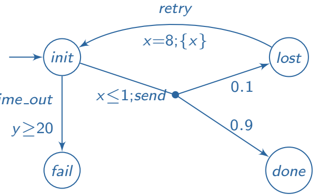
6 Discrete-time Markov chains
- Reachability properties
- Branching-time model checking
- Linear-time model checking

7 Markov decision processes
- Reachability properties
- Branching-time model checking
- Linear-time model checking

8 Probabilistic timed automata

9 Continuous-time Markov chains

Probabilistic timed automata



Probabilistic timed automata (PTA)

A probabilistic timed automaton is a tuple $\mathcal{P} = (L, \ell_0, Act, \mathcal{X}, E, \mathsf{lab}, AP)$ with

- L finite set of locations and $\ell_0 \in L$ the initial locations
- Act finite set of actions
- $\mathcal{X}$ finite set of clocks
- $E \subseteq L \times \mathcal{G} \times Act \times \mathsf{Dist}(2^{\mathcal{X}} \times L)$ set of probabilistic edges where $\mathcal{G}$ is the set of guards
- $\mathsf{lab} : L \to 2^{AP}$ labels locations with atomic propositions.

Semantics of PTA

The semantics of a PTA is an infinite-state Markov decision process $\mathcal{M} = (L \times \mathbb{R}_+^{\mathcal{X}}, Act \cup \mathbb{R}_+, \mathbf{P}, (\ell_0, 0), \mathrm{lab}, AP)$.

- States: $L \times \mathbb{R}_+^{\mathcal{X}}$ composed of a location and a valuation
- Actions: $Act \cup \mathbb{R}_+$ partitionned into discrete actions and delays
- Probabilistic transition function: $\mathbf{P}$ such that

  time transition for any delay $\tau \in \mathbb{R}_+$, there are deterministic transitions $(\ell, v) \xrightarrow{\tau} (\ell, v+\tau)$: $\mathbf{P}((\ell, v), \tau, (\ell, v+\tau)) = 1$.

  discrete transition for every edge $(\ell, g, \alpha, \delta)$, and any state (ℓ, v) with $v \models g$, there are probabilistic transitions on α from (ℓ, v):

$$\mathbf{P}((\ell, v), \alpha, (\ell', v')) = \sum_{Y \subseteq \mathcal{X} \mid v_{[Y \leftarrow 0]} = v'} \delta(Y, \ell')$$

- Labelling: $\mathrm{lab}((\ell, v)) = \mathrm{lab}(\ell)$.

Reachability analysis

Goal: Compute min/max probability of reaching a set of target locations.

Region graph MDP $\mathcal{M}_\mathcal{R}$

- States: $L_\mathcal{R} = L \times \mathcal{R}$.
- Actions: $Act_\mathcal{R} = Act \cup \mathsf{Succ}$: discrete action or time-successor.
- Transitions: $\mathbf{P}_\mathcal{R}$
  - $\mathbf{P}_\mathcal{R}((\ell, R), \mathsf{Succ}, (\ell, R')) = 1$ for R' the direct time-successor of R
  - for every state (ℓ, R) and every edge $(\ell, g, \alpha, \delta)$ with $R \models g$, $\mathbf{P}_\mathcal{R}((\ell, R), \alpha, (\ell', R')) = \sum_{Y \subseteq \mathcal{X} \mid R' = R_{[Y \leftarrow 0]}} \delta(Y, \ell')$.
- Labelling: $\mathsf{lab}_\mathcal{R}(\ell, R) = \mathsf{lab}(\ell)$

Correction of the region graph MDP

$$Pr^{\max}_{\mathcal{P}}((\ell_0, 0) \models \Diamond T) = Pr^{\max}_{\mathcal{M}_\mathcal{R}}((\ell_0, 0) \models \Diamond (T \times \mathcal{R}))$$
$$Pr^{\min}_{\mathcal{P}}((\ell_0, 0) \models \Diamond T) = Pr^{\min}_{\mathcal{M}_\mathcal{R}}((\ell_0, 0) \models \Diamond (T \times \mathcal{R}))$$

Also works for invariants and time-bounded reachability properties.

Beyond reachability: PTCTL

Syntax of PTCTL

$\psi ::= \mathtt{tt} \mid a \mid g \mid \psi_1 \wedge \psi_2 \mid \neg\psi \mid \mathbb{P}_J(\psi_1 \mathsf{U} \psi_2) \mid \mathbb{P}_J(\psi_1 \mathsf{U}^I \psi_2)$

$\mathcal{Z}$ set of formula clocks, $g \in \mathcal{G}$ zone over $\mathcal{X} \cup \mathcal{Z}$, $J \subseteq [0,1]$ interval, $I \subseteq \mathbb{R}_+$ interval with integer bounds

Examples

- $\mathbb{P}_{>0}(\Diamond^{[1,2]}\mathtt{error})$
- $\mathbb{P}_{=1/2}(\mathtt{on}\ \mathsf{U}\ \mathbb{P}_{\geq 0.9}(\Diamond z \leq 2 \wedge \neg\mathtt{on}))$

Semantics

- $s \models \mathbb{P}_J(\varphi)$ iff $\forall\sigma,\ Pr^\sigma(\{\rho = s\cdots \mid \rho \models \varphi\}) \in J$
- for time-divergent run $\rho = (\ell_0, v_0) \xrightarrow{\tau_1, a_1} (\ell_1, v_1) \xrightarrow{\tau_2, a_2} \cdots$
  $\rho \models \psi_1 \mathsf{U}^I \psi_2$ if and only if there exists $i \geq 0$, there exists $\tau \in [0, \tau_i]$ such that
  - $(\ell_i, v_i + \tau) \models \psi_2$ with $\sum_{k=1}^{i} \tau_k + \tau \in I$,
  - $\forall j \leq i, \forall \tau' \in [0, \tau_j]$,
    $\sum_{k=1}^{j} \tau_k + \tau' \leq \sum_{k=1}^{i} \tau_k + \tau \implies (\ell_j, v_j + \tau') \models \psi_1 \vee \psi_2$

Model checking PTCTL on PTA

PTCTL model checking on PTA

Input: PTA $\mathcal{P}$ with clocks $\mathcal{X}$, and PTCTL formula ψ with clocks $\mathcal{Z}$
Question: $\mathcal{P} \models \psi$?

- build a region graph MDP $\mathcal{M}_{\mathcal{R}}(\psi)$ on $\mathcal{X} \cup \mathcal{Z}$
- translate ψ into a PCTL formula ψ' such that
  ψ holds in a state of $\mathcal{P}$ iff ψ' holds in the corresponding state of $\mathcal{M}_{\mathcal{R}}(\psi)$
- decide whether $\mathcal{M}_{\mathcal{R}}(\psi) \models \psi'$ using PCTL model checking for MDP

In practice: more efficient techniques (digital clocks, zone graph MDP, abstract MDP refined on demand)

Outline

Continuous-time Markov chains

Continuous-time Markov chains (CTMC)

$\mathcal{M} = (S, \mathbf{R}, p_{\text{init}}, \text{lab}, AP)$ with

- S finite set of states, $\mathbf{R}$ rate matrix, p_{init} initial distribution.

$\mathbf{R}(s,t) = \lambda$ means that the average time when going from s to t is $\frac{1}{\lambda}$
$E(s) = \sum_{t \in S} \mathbf{R}(s,t)$ is the exit rate of s.

Interpretation

- probability that $s \xrightarrow{\lambda} t$ is enabled in $[0,\tau]$: $1 - e^{-\lambda\tau}$;
- probability that $s \xrightarrow{\lambda} t$ is taken in $[0,\tau]$: $\frac{R(s,t)}{E(s)}(1 - e^{-E(s)\tau})$;
- probability to leave s in $[0,\tau]$: $(1 - e^{-E(s)\tau})$.

Example: Triple modular redundancy

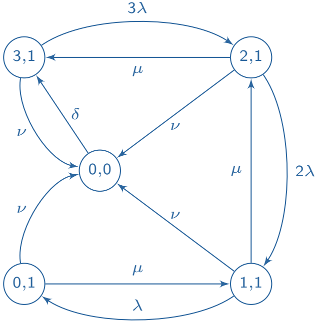
Fault-tolerant majority voting system: 3 processors and a single voter.

- failure rate of processor (resp. voter): λ (resp. ν)
- repair rate of processor (resp. voter): μ (resp. δ)

$S = \{(0,0),(0,1),(1,1),(2,1),(3,1)\}$

$$\mathbf{R} = \begin{pmatrix} 0 & 0 & 0 & 0 & \delta \\ \nu & 0 & \mu & 0 & 0 \\ \nu & \lambda & 0 & \mu & 0 \\ \nu & 0 & 2\lambda & 0 & \mu \\ \nu & 0 & 0 & 3\lambda & 0 \end{pmatrix}$$

$E = (\delta\ ,\ \nu+\mu\ ,\ \nu+\lambda+\mu\ ,\ \nu+2\lambda+\mu\ ,\ \nu+3\lambda)$

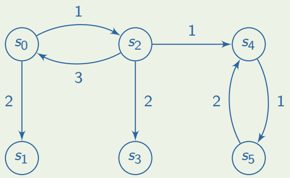


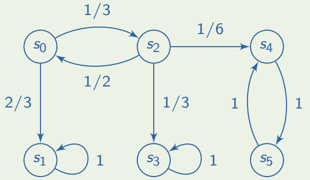
Embedded DTMC

Given $(S, \mathbf{R}, p_{\text{init}})$ a CTMC, its embedded DTMC is $(S, \mathbf{P}, p_{\text{init}})$ with

$$\mathbf{P}(s,t) = \begin{cases} \frac{\mathbf{R}(s,t)}{E(s)} & \text{if } E(s) > 0 \\ 0 & \text{otherwise} \end{cases}$$

Example





Elementary probabilities in CTMC

Transient probability vector

$\pi(t) = (\pi_1(t), \cdots \pi_n(t))$ for $t \geq 0$, where $\pi(t)$ is the probability to be in state s_i at time t.

Computed by solving a system of linear differential equations:

$$\pi'(t) = \pi(t)\big(\mathbf{R} - \mathrm{diag}(\mathrm{E})\big).$$

Steady-state probability vector

$\pi = (\pi_1, \cdots, \pi_n)$, with $\pi_i = \lim_{t\to\infty} \pi_i(t)$.

If $\mathcal{M}$ has a single BSCC, π is computed by solving a system of linear equations:

$$\pi\big(\mathbf{R} - \mathrm{diag}(\mathrm{E})\big) = 0 \text{ and } \sum_i \pi_i = 1.$$

Otherwise, see computation of $Sat(\mathbb{S}_J(\psi))$, Slide 91.

Continuous Stochastic Logic

Syntax of CSL

- state formulae: $\psi ::= \mathtt{tt} \mid a \mid \neg\psi \mid \psi_1 \wedge \psi_2 \mid \mathbb{S}_J(\psi) \mid \mathbb{P}_J(\varphi)$
- path formulae: $\varphi ::= \bigcirc_I \psi \mid \psi_1 \mathsf{U}_I \psi_2$

$s \models \mathbb{S}_J(\psi)$ iff the probability that ψ holds in steady state lies in J.

Example

In at least 90% of the cases, a goal state is reached within 3 time units guaranteeing 0.99 long-run availability.

$$\mathbb{P}_{[0.9,1]}(s_0 \models \mathtt{tt}\ \mathsf{U}_{[0,3]}\ \mathbb{S}_{[0.99,1]}\ \mathtt{goal})$$

Model checking CSL

Recursive computation of $Sat(\psi)$.

- Nonstochastic part: as for CTL
- Probabilistic formulae without time bounds: as for PCTL
  - on the embedded DTMC
- To do: stochastic timed operators.

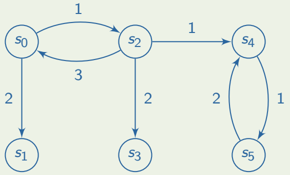
Steady-state operator: $\mathbb{S}_J(\psi)$

- For a strongly connected CTMC

$$s \in Sat(\mathbb{S}_J(\psi)) \text{ if and only if } \sum_{t \in Sat(\psi)} \pi_t \in J.$$

- For an arbitrary CTMC
  - determine the set of BSCC $\mathcal{C}$,
  - for $C \in \mathcal{C}$ compute
    - the probability to reach C: $Pr(s_0 \models \Diamond C)$
    - the steady-state probability of ψ-states: π_t for $t \in C \cap Sat(\psi)$
  - $s \in Sat(\mathbb{S}_J(\psi))$ if and only if $\sum_C \Big(Pr(s_0 \models \Diamond C) \cdot \sum_{t \in C \cap Sat(\psi)} \pi_t \Big) \in J$.

Example



$\mathrm{lab}(s_1) = \mathrm{lab}(s_2) = \mathrm{lab}(s_5) = \{a\}$
$\mathrm{lab}(s_0) = \mathrm{lab}(s_3) = \mathrm{lab}(s_4) = \emptyset$

Does $s_0 \models \mathbb{S}_{>0.8} a$?

Time-bounded reachability: $\psi_1 \mathsf{U}_{\leq \tau} \psi_2$

$Pr(s \models \psi_1 \mathsf{U} \psi_2)$ is the least fixpoint of the following system

- If $s \models \psi_1 \wedge \neg \psi_2$, then

$$Pr(s \models \psi_1 \mathsf{U}_{\leq \tau} \psi_2) = \int_0^{\tau} \sum_{t \in S} \underbrace{\mathbf{P}(s,t) E(s) e^{-E(s)x}}_{\substack{\text{probability to move to} \\ t \text{ at time } x}} \underbrace{Pr(t \models \psi_1 \mathsf{U}_{\leq \tau - x} \psi_2)}_{\substack{\text{probability to fulfill } \psi_1 \mathsf{U} \psi_2 \\ \text{before time } \tau - x \text{ from } s'}} \, \mathrm{d}x.$$

- If $s \models \psi_2$, then $Pr(s \models \psi_1 \mathsf{U}_{\leq t} \psi_2) = 1$.
- Else, $Pr(s \models \psi_1 \mathsf{U}_{\leq t} \psi_2) = 0$.

Tools in a nutshell

PRISM

- developed at Oxford University
- model checking of DTMC, MDP, PTA and CTMC

http://www.prismmodelchecker.org/

MRMC

- developed at RWTH Aachen University
- model checking of DTMC, CTMC

http://www.mrmc-tool.org/

References

- Mathematical foundations
  - Kemeny, Snell and Knapp. Denumerable Markov chains. D. Van Nostrand Co, 1966.
  - Puterman. Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons, 1994.
- Probabilisic model checking
  - Baier and Katoen. Principles of model checking. MIT Press, 2008.
  - Baier, Haverkort, Hermanns and Katoen. Model checking algorithms for continuous-time Markov chains. IEEE Transactions on Software Engineering, 2003.