
UNITED STATES

What we have seen so far

Model Construction Basics: Parallel Composition

Discrete Time Markov Chains

Probabilistic CTL Model Checking

Recall:

Holger Hermanns

Bisimulation

$\mathcal{T}_1 = (S_1, \mathrm{Act}_1, \longrightarrow_1, \ldots)$ $\qquad$ $\mathcal{T}_2 = (S_2, \mathrm{Act}_2, \longrightarrow_2, \ldots)$

A relation $\mathbf{R} \subseteq S_1 \times S_2$ is a bisimulation, if

for all $(s_1, s_2) \in \mathbf{R}$ and for all $\alpha \in \mathrm{Act}$:

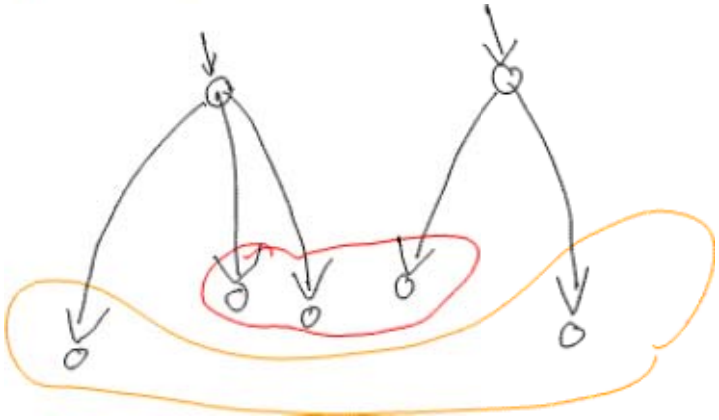
(1) $s_1 \xrightarrow{\alpha}_1 s_1'$ implies $\exists\, s_2 \xrightarrow{\alpha}_2 s_2'$ such that $(s_1', s_2') \in \mathbf{R}$

(2) $s_2 \xrightarrow{\alpha}_2 s_2'$ implies $\exists\, s_1 \xrightarrow{\alpha}_1 s_1'$ such that $(s_1', s_2') \in \mathbf{R}$

Bisimulation equivalence of $\mathcal{T}_1$ and $\mathcal{T}_2$ requires that

$\mathcal{T}_1$ and $\mathcal{T}_2$ can *simulate each other* in a stepwise manner

$\mathcal{T}_1 \sim \mathcal{T}_2$ iff there is a bisimulation $\mathbf{R}$ for $(\mathcal{T}_1, \mathcal{T}_2)$ relating the initial states.



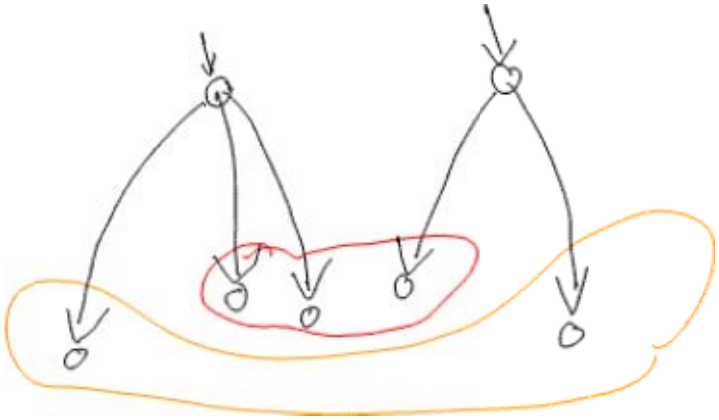
Bisimulation for DTMCs

$\mathcal{D}_1 = (S, \mathbf{P}, \ldots)$ (now with state labels)

An equvialence relation $\mathbf{R} \subseteq S \times S$ is a bisimulation, if for all $(s_1, s_2) \in \mathbf{R}$:

1. $L(s_1) = L(s_2)$,
2. $\mathbf{P}(s_1, C) = \mathbf{P}(s_2, C)$ for each equivalence class C of $\mathbf{R}$.

Two states $s_1, s_2 \in S$ are bisimilar $(s_1 \sim s_2)$ if there is a bisimulation $\mathbf{R}$ with $(s_1, s_2) \in \mathbf{R}$.



PCTL-equivalence and $\sim$ agree.

How to decorate a DTMC with actions? (1)

So far: A DTMC is a tuple: $(S, \mathbf{P}, \pi(0), \ldots)$ where

- S is the set of states,
- $\mathbf{P} : S \times S \to [0,1]$ with $\sum_{s' \in S} \mathbf{P}(s,s') = 1$ is the transition matrix,
- $\pi(0)$ is the initial distribution.

How about:

An action-labelled DTMC is a tuple: $(S, \mathbf{P}, \pi(0), \ldots)$ where

- S is the set of states,
- $\mathbf{P} : S \times \mathrm{Act} \times S \to [0,1]$ with $\sum_{\alpha, s'} \mathbf{P}(s,\alpha,s') = 1$ is the transition matrix,
- $\pi(0)$ is the initial distribution.

Recall:

Synchronous product

$\mathcal{T}_1 = (S_1, \mathsf{Act}_1, \longrightarrow_1, \ldots)$ $\qquad$ $\mathcal{T}_2 = (S_2, \mathsf{Act}_2, \longrightarrow_2, \ldots)$

The synchronous product $\mathcal{T}_1 \otimes \mathcal{T}_2$ is:

$$\mathcal{T}_1 \otimes \mathcal{T}_2 \;=\; (S_1 \times S_2, \mathsf{Act}, \longrightarrow, \ldots)$$

where the transition relation $\longrightarrow$ is given by:

$$\frac{s_1 \xrightarrow{\alpha}_1 s_1' \;\wedge\; s_2 \xrightarrow{\beta}_2 s_2'}{\langle s_1, s_2 \rangle \xrightarrow{\alpha * \beta} \langle s_1', s_2' \rangle}$$

action set Act is given by a function

$$* : \mathsf{Act}_1 \times \mathsf{Act}_2 \longrightarrow \mathsf{Act}, \qquad (\alpha, \beta) \mapsto \alpha * \beta$$

for parallel systems with fully synchronous execution

Synchronous product

$\mathcal{D}_1 = (S_1, \mathsf{Act}_1, \mathbf{P}_1, \ldots)$ $\qquad$ $\mathcal{D}_2 = (S_2, \mathsf{Act}_2, \mathbf{P}_2, \ldots)$

The synchronous product $\mathcal{D}_1 \otimes \mathcal{D}_2$ is:

$$\mathcal{D}_1 \otimes \mathcal{D}_2 \;=\; (S_1 \times S_2, \mathsf{Act}, \longrightarrow, \ldots)$$

where the *probability matrix* $\mathbf{P}$: is given by:

$$\frac{\mathbf{P}_1(s_1, \alpha, s_1') = p > 0 \;\wedge\; \mathbf{P}_2(s_2, \beta, s_2') = q > 0}{\mathbf{P}(\langle s_1, s_2\rangle, \alpha * \beta, \langle s_1', s_2'\rangle) \;=\; pq}$$

for parallel DTMCs with fully synchronous execution

Recall:

Interleaving operator

$\mathcal{T}_1 = (S_1, \mathrm{Act}_1, \longrightarrow_1, s_{01}, AP_1, L_1)$ $\quad$ $\mathcal{T}_2 = (S_2, \mathrm{Act}_2, \longrightarrow_2, s_{02}, AP_2, L_2)$

The composite transition system $\mathcal{T}_1 \,|||\, \mathcal{T}_2$ is:

$$\mathcal{T}_1 \,|||\, \mathcal{T}_2 \;=\; (S_1 \times S_2,\ \mathrm{Act}_1 \cup \mathrm{Act}_2,\ \longrightarrow,\ \langle s_{01}, s_{02}\rangle, AP, L)$$

where the transition relation $\longrightarrow$ is given by:

$$\frac{s_1 \xrightarrow{\alpha}_1 s_1'}{\langle s_1, s_2\rangle \xrightarrow{\alpha} \langle s_1', s_2\rangle} \qquad \frac{s_2 \xrightarrow{\alpha}_2 s_2'}{\langle s_1, s_2\rangle \xrightarrow{\alpha} \langle s_1, s_2'\rangle}$$

does not extend in an intuitive way to the DTMC setting!

How to decorate a DTMC with actions? (2)

So far: A DTMC is a tuple: $(S, \mathbf{P}, \pi(0), \ldots)$ where

- S is the set of states,
- $\mathbf{P} : S \times S \to [0,1]$ with $\sum_{s' \in S} \mathbf{P}(s,s') = 1$ is the transition matrix,
- $\pi(0)$ is the initial distribution.

Instead:

- S is the set of states,
- $\longrightarrow \subseteq S \times \mathrm{Act} \times (S \to [0,1])$, a probabilistic transition relation with $s \xrightarrow{\alpha} \mathbf{P}$ implies $\sum_{s'} \mathbf{P}(s') = 1$.
- $\pi(0)$ is the initial distribution.

This is the model of probabilistic automata, coined by Roberto Segala.

Interleaving operator for probabilistic automata

$$\mathcal{D}_1 = (S_1, Act_1, \longrightarrow_1, s_{01}, \ldots)$$
$$\mathcal{D}_2 = (S_2, Act_2, \longrightarrow_2, s_{02}, \ldots)$$

The composite transition system $\mathcal{D}_1 \,|||\, \mathcal{D}_2$ is given by:

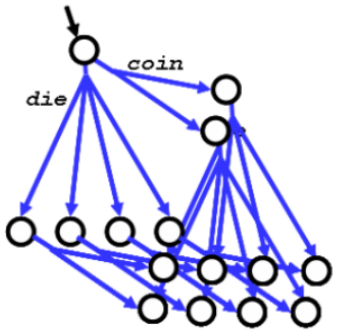
$$\mathcal{D}_1 \,|||\, \mathcal{D}_2 = (S_1 \times S_2,\ Act_1 \cup Act_2,\ \longrightarrow,\ \langle s_{01}, s_{02}\rangle,\ AP, L)$$

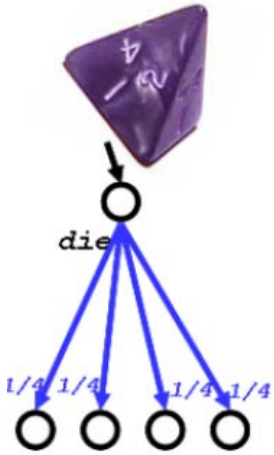
where the transition relation $\longrightarrow$ is given by:

$$\frac{s_1 \xrightarrow{\alpha}_1 \mathbf{P}_1}{\langle s_1, s_2\rangle \xrightarrow{\alpha} \mathbf{Q}_1} \qquad \frac{s_2 \xrightarrow{\alpha}_2 \mathbf{P}_2}{\langle s_1, s_2\rangle \xrightarrow{\alpha} \mathbf{Q}_2}$$

where $\mathbf{Q}_1(\langle s_1', s_2'\rangle) = \mathbf{P}_1(s_1')$ if $s_2' = s_2$ and 0 otherwise,

and $\mathbf{Q}_2(\langle s_1', s_2'\rangle) = \mathbf{P}_2(s_2')$ if $s_1' = s_1$ and 0 otherwise.





Synchronous message passing for probabilistic automata

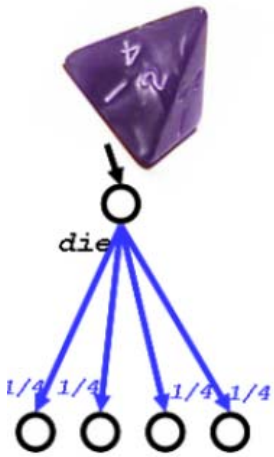
concurrent execution with **synchronization** over all actions in Syn

$\mathcal{D}_1 = (S_1, Act_1, \rightarrow_1, \dots)$ $\qquad$ $\mathcal{D}_2 = (S_2, Act_2, \rightarrow_2, \dots)$

$$\mathcal{D}_1 \parallel_{Syn} \mathcal{D}_2 = (S_1 \times S_2, Act_1 \cup Act_2, \rightarrow, \dots)$$

interleaving for every action $\alpha \in Act_i \setminus Syn$:

$$\frac{s_1 \xrightarrow{\alpha}_1 \mathbf{P}_1}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \mathbf{Q}_1} \qquad \frac{s_2 \xrightarrow{\alpha}_2 \mathbf{P}_2}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \mathbf{Q}_2}$$



handshaking (rendezvous) for $\alpha \in Syn$:

$$\frac{s_1 \xrightarrow{\alpha}_1 \mathbf{P}_1 \quad \wedge \quad s_2 \xrightarrow{\alpha}_2 \mathbf{P}_2}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \mathbf{P}_1\mathbf{P}_2}$$

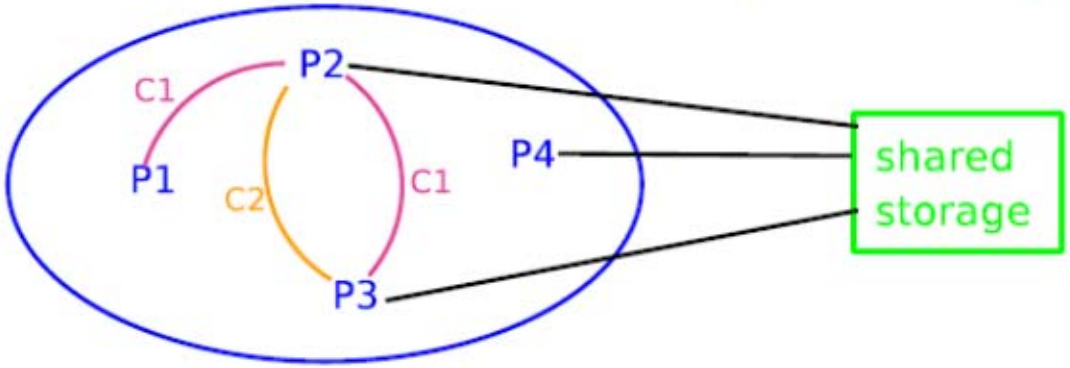
where $\mathbf{P}_1\mathbf{P}_2(\langle s_1', s_2' \rangle) = \mathbf{P}_1(s_1')\mathbf{P}_2(s_2')$.

Recall:

Channel systems and shared variable systems

We want to represent data-dependent concurrent systems with

- communication over **shared variables**
- **synchronous message passing** (channels of **capacity 0**)
- **asynchronous message passing** (**capacity** $\geq$ **1**)



This can all be encoded into

- transition systems

  and synchronous message passing

How to decorate a DTMC with actions? (3)

So far: A DTMC is a tuple: $(S, \mathbf{P}, \pi(0), \ldots)$ where

- S is the set of states,
- $\mathbf{P} : S \times S \to [0,1]$ with $\sum_{s' \in S} \mathbf{P}(s,s') = 1$ is the transition matrix,
- $\pi(0)$ is the initial distribution.

Instead:

- S is the set of states,
- $\mathbf{P} : S \times S \to [0,1]$ with $\sum_{s'} \mathbf{P}(s,s') = 1$, the transition matrix,
- $\longrightarrow \subseteq S \times \mathrm{Act} \times S$, a labelled transition relation,
- $\pi(0)$ is the initial distribution.

This model goes back to Hans Hansson, with some twisting it is equipotent.

Probabilistic Automata . . . Markov Decision Processes

Ok. What is the relation to Markov chains?

Well, if we fix a resolution of the non-determinism we get a DTMC.

How do we fix?

However you like!

But, someone must decide which $(s, a, \mathbf{P}) \in \longrightarrow$ *to pick in state* s.

True. This is what an adversary (policy or scheduler) is good for.

- An adversary is a function $A : \mathit{Paths}_{\mathit{fin}} \to \mathit{Distr}(\mathrm{Act} \times \mathit{Distr}(S))$
  such that $A(\sigma)((a, \mathbf{P})) > 0 \Rightarrow \mathit{last}(\sigma) \xrightarrow{a} \mathbf{P}$
- It maps the entire history
  to a distribution over possible choices $(a, \mathbf{P})$ in the present state.

Adv denotes the set of all adversaries.

Note: The induced DTMC is an infinite object – states are paths.

PCTL revisited

Syntax

State formulas:

$$\Phi := \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \mathbb{P}_J(\phi)$$

where $a \in AP$, $J \subseteq [0,1]$ is an interval with rational bounds.

Path formulas:

$$\phi := \mathcal{X}\Phi \mid \Phi_1 \,\mathcal{U}\, \Phi_2 \mid \Phi_1 \,\mathcal{U}^{\leq n}\, \Phi_2$$

where $n \in \mathbb{N}$.

Semantics

Satisfaction relation for PCTL path formulas

Unchanged.

Satisfaction relation for PCTL state formulas

Given an MDP $\mathcal{M} = (S, \longrightarrow, \pi(0), L)$, state $s \in S$,
the satisfaction relation $\models$ is defined by:

- $s \models a$ iff $a \in L(s)$,
- $s \models \neg\Phi$ iff $s \not\models \Phi$,
- $s \models \Phi \wedge \Psi$ iff $s \models \Phi$ and $s \models \Psi$,
- $s \models \mathbb{P}_J(\phi)$ iff $\mathbf{Pr}_s^{ind(\mathcal{M},A)}(\phi) \in J$ for all adversaries A

Again, PCTL path formulas are measurable.

What adversary is really needed for what type of property?

- Random selection does not add anything.
- For bounded until, adversaries better count steps, not more. $\mathcal{U}^{\leq n}$
- For unbounded until, adversaries without history suffice. $\mathcal{U}$

Recall:

Fixed point characterisation for DTMC

We define:

$$S_{=1} = \{s \mid \mathbf{Pr}_s(C \,\mathcal{U}\, B) = 1\}$$
$$S_{=0} = \{s \mid \mathbf{Pr}_s(C \,\mathcal{U}\, B) = 0\}$$
$$S_{=?} = S \setminus (S_{=1} \cup S_{=0})$$

and these sets are obtained via the underlying graph!

Theorem

The vector $(\mathbf{Pr}_s(C \,\mathcal{U}\, B))_{s \in S}$ *is the unique fixed point of the operator* $\nabla : (S \to [0,1]) \to (S \to [0,1])$ *defined by:*

$$(\nabla(\mathbf{x}))_s = \begin{cases} 1 & \text{if } s \in S_{=1} \\ 0 & \text{if } s \in S_{=0} \\ \sum_{t \in S} \mathbf{P}(s,t)\, x_t & \text{else} \end{cases}$$

Fixed point characterisation for PA

We define:

$$S_{=1} = B$$
$$S_{=0} = \{s \mid \forall A : \mathbf{P}_s^{ind(\mathcal{M},A)}(C \ \mathcal{U} \ B) = 0\}$$
$$S_{=?} = S \setminus (S_{=1} \cup S_{=0})$$

and these sets are obtained via the underlying graph!

Theorem

The vector $(\mathbf{Pr}_s^{\max}(C \ \mathcal{U} \ B))_{s \in S}$ *is the least fixed point of the operator* $\nabla : (S \to [0,1]) \to (S \to [0,1])$ *defined by:*

$$(\nabla(\mathbf{x}))_s = \begin{cases} 1 & \text{if } s \in S_{=1} \\ 0 & \text{if } s \in S_{=0} \\ \max_{(s,a,\mathbf{P}) \in \to} \sum_{t \in S} \mathbf{P}(t)\, x_t & \text{else} \end{cases}$$

and similar for $\mathbf{Pr}^{\min}$

(instances of Bellmann equations)

Putting things together: PCTL model checking for PAs

Given a probabilistic automaton $(S, \longrightarrow, \pi(0), L)$ and a PCTL formula Φ:
We determine $Sat(\Phi)$ as follows.

Bottom-up parse-tree traversal (Obvious for *true*, a, $\Phi_1 \wedge \Phi_2$, $\neg\Phi$.)

For $\mathbb{P}_{[l,u]}(\mathcal{X}\Phi)$, $\mathbb{P}_{[l,u]}(\Phi_1\ \mathcal{U}\ \Phi_2)$, $\mathbb{P}_{[l,u]}(\Phi_1\ \mathcal{U}^{\leq n}\ \Phi_2)$:

in all cases, compute extremal probabilities $\mathbf{Pr}^{\min}$ and $\mathbf{Pr}^{\max}$

- $\mathcal{X}\Phi$: you work it out!
- $\Phi_1\ \mathcal{U}^{\leq n}\ \Phi_2$: apply ∇ n times.
- $\Phi_1\ \mathcal{U}\ \Phi_2$: apply ∇ until convergence (value iteration)
  or use linear programming, or ....

and then check (statewise) whether $\mathbf{Pr}^{\min} \geq l$ and $\mathbf{Pr}^{\max} \leq u$.

Complexity

- Overall complexity:

  polynomial in the size of $\mathcal{D}$,

  linear in the size of Φ,

  linear in the maximal step bound n.

Bisimulation

. . . can be lifted to probabilistic automata, such that:

State labels: PCTL-equivalence and bisimulation agree.

Transition labels: Bisimulation is a congruence for $\|_{\mathsf{Syn}}$.

Both: Bisimulation is a congruence for $\|_{\mathsf{Syn}}$ and implies PCTL-equivalence.

PRISM Case Studies

PRISM has been used to analyse a wide range of case studies in many different application domains. Below you can find more information and about a large number of these. Typically, you can find descriptions of the case study and its model(s), PRISM language source code and experimental results.

We are always happy to include details of externally developed case studies. If you would like to contribute content about your work with PRISM, or you want us to add a pointer to a publication about your PRISM-related work, please contact us.

Randomised distributed algorithms

These case studies examine the correctness and performance of various *randomised distributed algorithms* taken from the literature.

- Randomised self-stabilising algorithms (Herman) (Israeli & Jalfon) (Beauquier et al.)
- Randomised two process wait-free test-and-set (Tromp & Vitanyi)
- Synchronous leader election protocol (Itai & Rodeh)
- Asynchronous leader election protocol (Itai & Rodeh)
- Randomised dining philosophers (Lehmann & Rabin)
- Randomised dining philosophers (Lynch, Saias & Segala)
- Dining cryptographers (Chaum)
- Randomised mutual exclusion (Rabin)
- Randomised mutual exclusion (Pnueli & Zuck)
- Randomised consensus protocol (Aspnes & Herlihy) (with Cadence SMV and PRISM) (See also [KNS01a])

CTMC

DTMC

MDP