Model Checking Probabilistic Systems

Joost-Pieter Katoen

Software Modeling and Verification Group

affiliated to University of Twente, Formal Methods and Tools

ARTIST2 Motives Winterschool, February 22, 2007

Probabilities help

- When analysing system performance and dependability
  - to quantify arrivals, waiting times, time between failure, QoS, ...
- When modelling uncertainty in the environment
  - to quantify environmental factors in decision support
  - to quantify unpredictable delays, express soft deadlines, ...
- When building protocols for networked embedded systems
  - randomized algorithms
- When analysing large populations
  - number of nodes in the internet, number of end-users, ...

Probability elsewhere

- In performance modelling (Erlang, 1907)
  - models: typically continuous-time Markov chains
  - emphasis on steady-state and transient measures

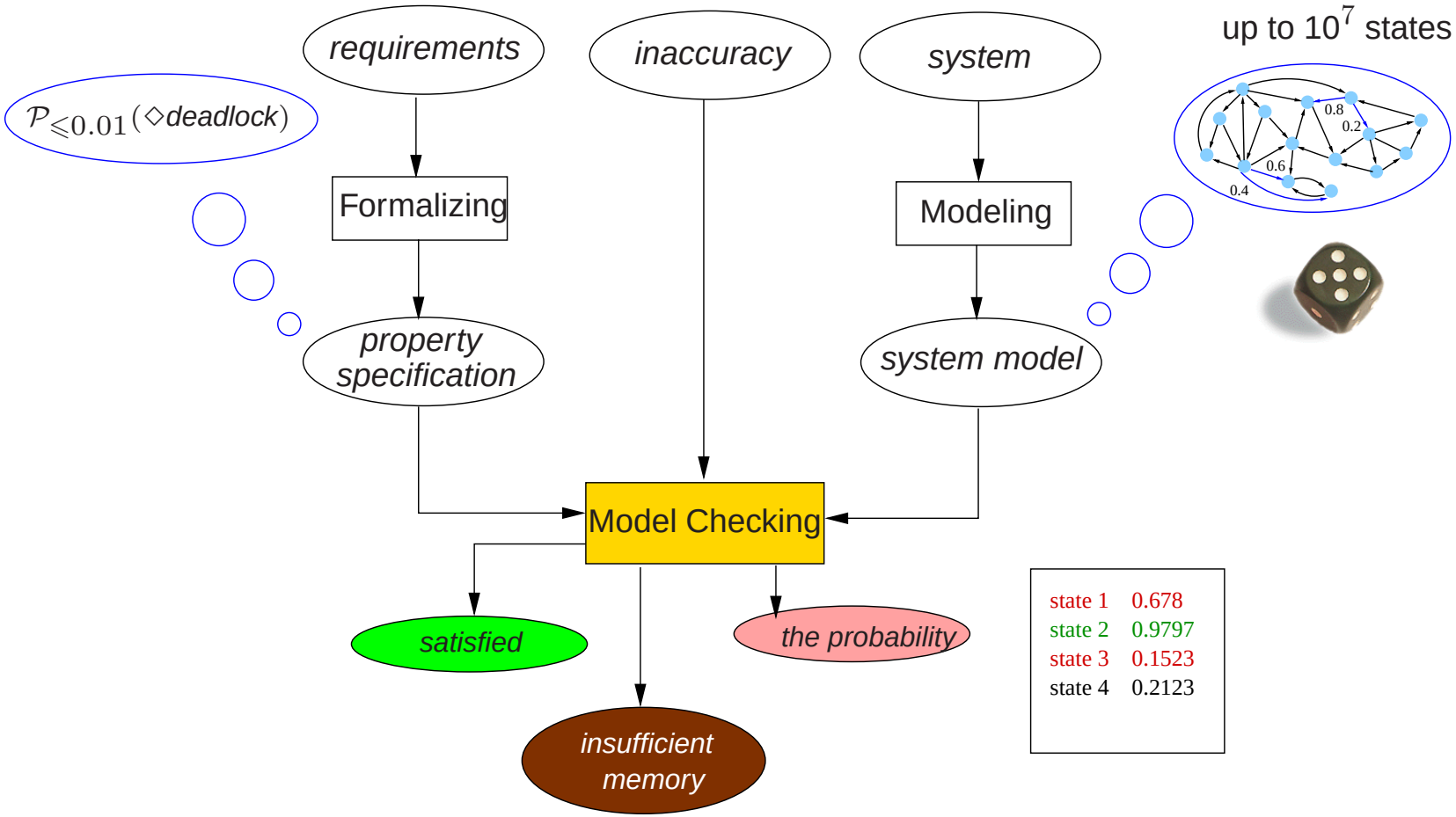
- In stochastic control theory and operations research (Bellman, 1957)
  - models: typically discrete-time Markov decision models
  - emphasis on finding optimal policies for average measures

- Our focus: model checking Markov chains
  - temporal logic $\Rightarrow$ unambiguous and precise *measure-specification*
  - model-checking techniques $\Rightarrow$ no expert algorithmic knowledge needed
  - complex (new) measures are concisely specified and *automatically verified*
  - exchanging techniques with the other two areas

Probabilistic verification so far

- Termination of probabilistic programs (Hart, Sharir & Pnueli, 1983)
  - does a probabilistic program terminate with probability one?
- Markov decision processes (Courcoubetis & Yannakakis, 1988)
  - does a certain (linear) temporal logic formula hold with probability p?
- Discrete-time Markov chains (Hansson & Jonsson, 1990)
  - can we reach a goal state via a given trajectory with probability p?
- Discrete-time Markov decision processes (Bianco & de Alfaro, 1995)
  - what is the maximal (or minimal) probability of doing this?
- Continuous-time Markov chains (Baier, Katoen & Hermanns, 1999)
  - can we do so within a given time interval I?

What is probabilistic model checking?



Characteristics

- What is inside?
  - temporal logics and model checking
  - numerical and optimisation techniques from performance and OR
- What can be checked?
  - time-bounded reachability, long-run averages, safety and liveness
- What is its usage?
  - powerful tools: PRISM (4,000 downloads), MRMC, Petri net tools, Probmela
  - applications: distributed systems, security, biology, quantum computing . . .

Characteristics

- What is inside?
  - temporal logics and model checking
  - numerical and optimisation techniques from performance and OR
- What can be checked?
  - time-bounded reachability, long-run averages, safety and liveness
- What is its usage?
  - powerful tools: PRISM (4,000 downloads), MRMC, Petri net tools, Probmela
  - applications: distributed systems, security, biology, quantum computing . . .
- What are its deficiencies?
  - limited (automated) abstraction techniques
  - limited diagnostic feedback in case of property refutation

Overview of the tutorial

- Introduction
- Discrete-time probabilistic models
  - fully probabilistic systems: discrete-time Markov chains
  - probabilistic CTL, verification algorithms, counterexamples
- Continuous-time probabilistic models
  - fully probabilistic systems: continuous-time Markov chains
  - stochastic CTL, verification algorithms, experimental results
  - exhibiting non-determinism: continuous-time MDPs
- Stochastic cost models (not today)
  - discrete-time and continuous time Markov reward models

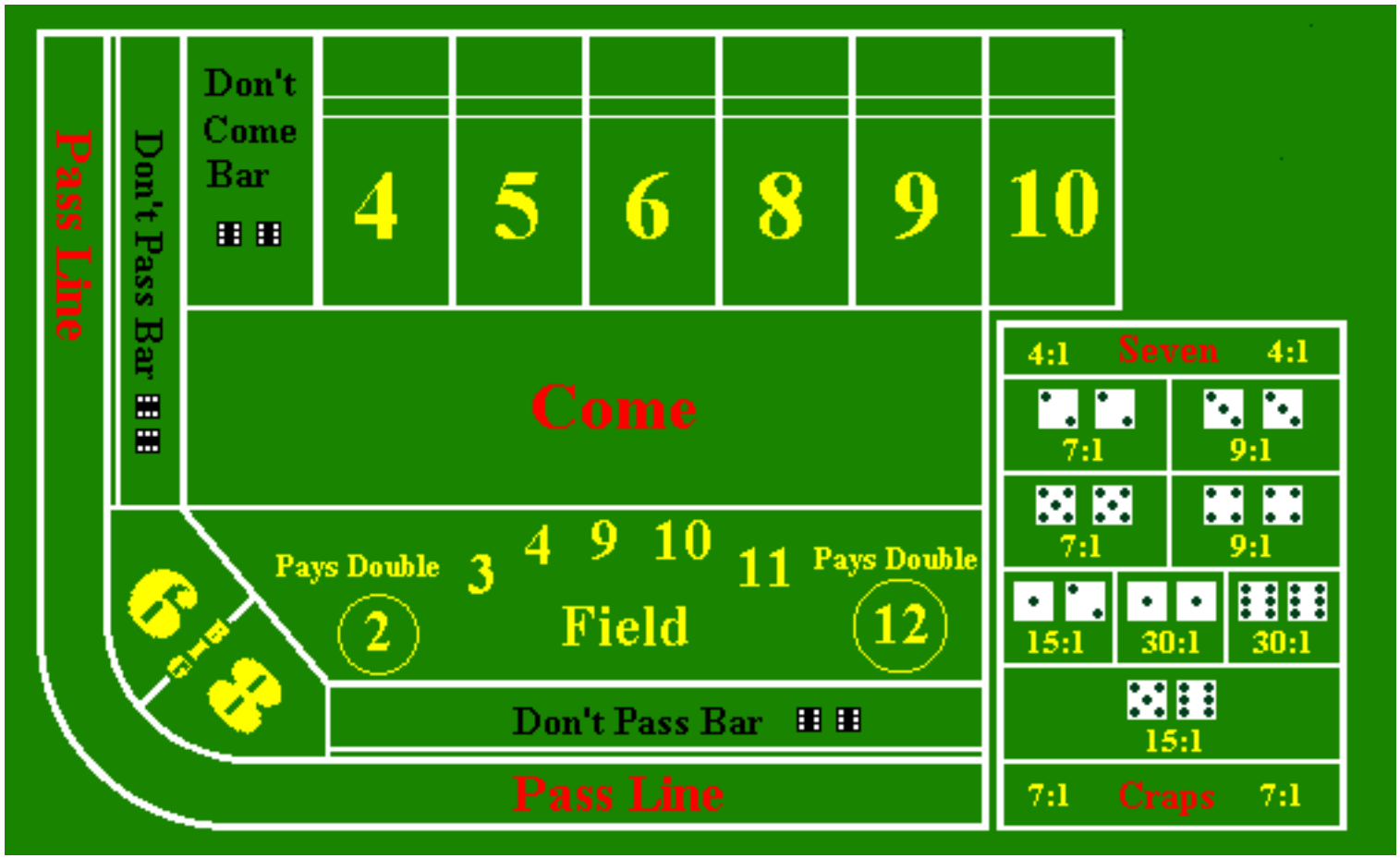
Finite labeled DTMCs

A labeled DTMC $\mathcal{D}$ is a triple $(S, \mathbf{P}, L)$ with:

- S, a finite set of states
- $\mathbf{P} : S \times S \to [0,1]$, a probability matrix with $\sum_{s'} \mathbf{P}(s,s') = 1$
  - $\mathbf{P}(s,s')$ is the probability to jump from s to s' in one step
  - s is absorbing if $\mathbf{P}(s,s) = 1$
- $L : S \to 2^{AP}$, the labelling function
  - $L(s)$ is the set of atomic propositions that are valid in s

$\Rightarrow$ a DTMC is a Kripke structure with only probabilistic transitions

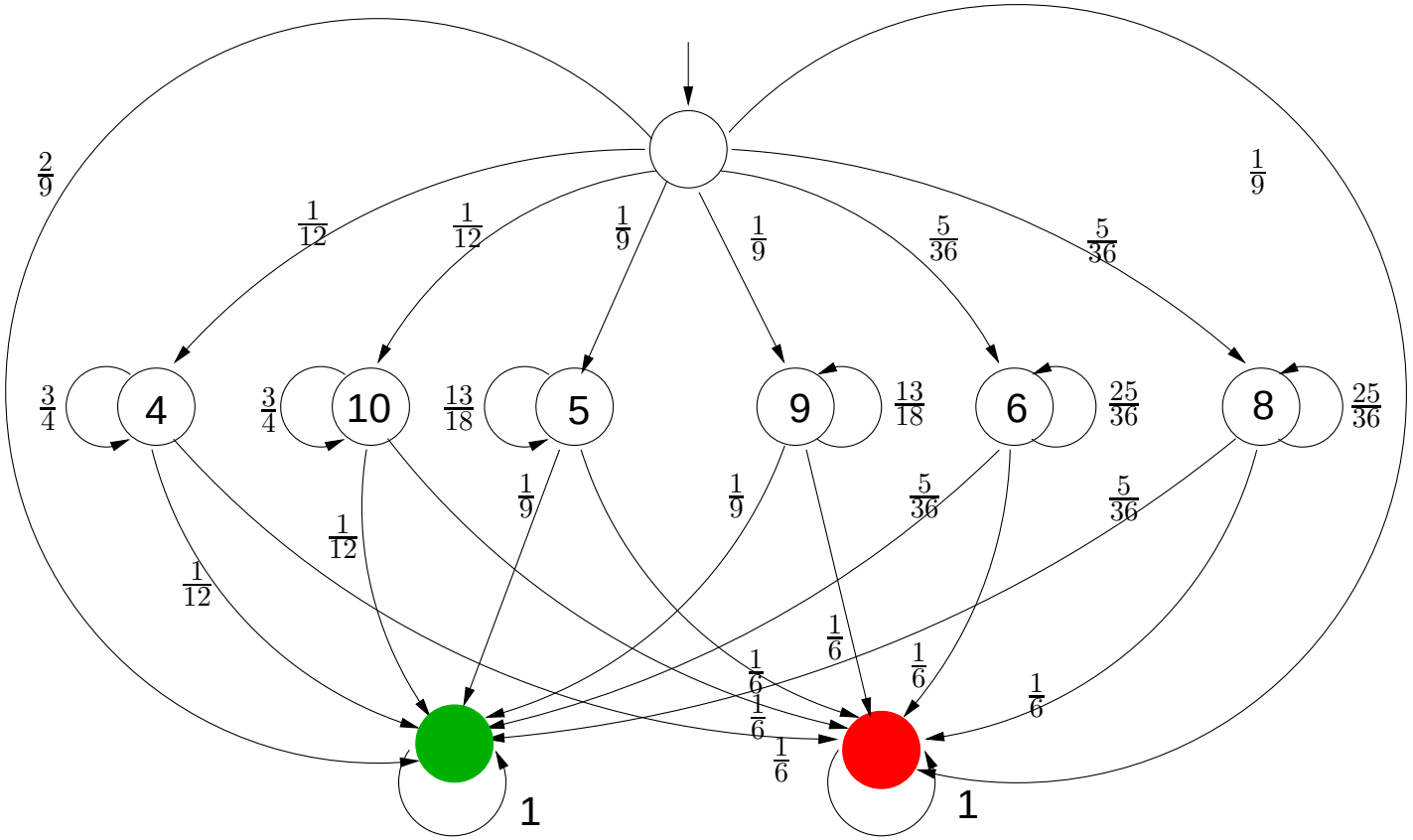
Craps


Pass Line
Don't Pass Bar
Don't Come Bar
4
5
6
8
9
10
Come
4:1 Seven 4:1
7:1
9:1
7:1
9:1
15:1
30:1
30:1
15:1
7:1 Craps 7:1
Pays Double
3
4 9 10 11
Pays Double
2
Field
12
Don't Pass Bar
Pass Line

Craps

- Roll two dice and bet on outcome
- Come-out roll (“pass line” wager):
  - outcome 7 or 11: win
  - outcome 2, 3, and 12: loss (“craps”)
  - any other outcome: roll again (outcome is “point”)
- Repeat until 7 or the “point” is thrown:
  - outcome 7: loss (“seven-out”)
  - outcome the point: win
  - any other outcome: roll again

A DTMC model of Craps

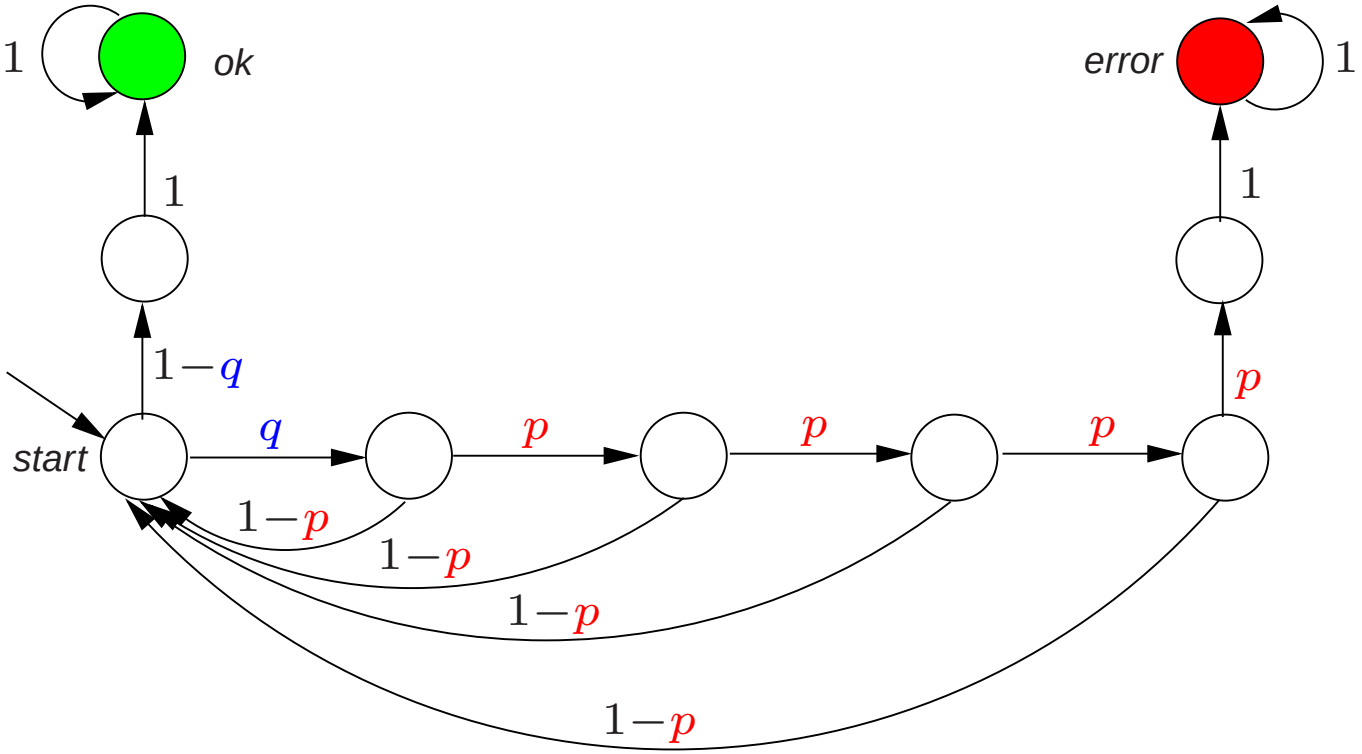


Address assignment in IPv4

- IPv4 is aimed at plug-and-play networks for domestic appliances
- New devices must get a unique IP address in an automated way
- This is done by the IPv4 zeroconf protocol (proposed by IETF)
  - randomly select one of the 65,024 possible addresses
  - loop: as long as # sent probes $< n$
    - broadcast probe “who is using this address?”
    - receive reply? $\rightarrow$ go back to initial step of protocol
    - receive no reply within $r > 0$ time units
      - # sent probes $= n$? $\rightarrow$ use selected address
      - # sent probes $< n$? $\rightarrow$ repeat loop

(Cheshire, Adoba & Guttman, 2001)

The probabilistic host behaviour



p = probability of message loss; q = probability of selecting occupied address

(Bohnenkamp *et al.*, 2001)

Probabilistic CTL (Hansson & Jonsson, 1990)

- For $a \in AP$, $J \subseteq [0,1]$ an interval with rational bounds, and natural n:

$$\Phi \;::=\; \text{true} \mid a \mid \Phi \wedge \Phi \mid \neg\Phi \mid \mathbb{P}_J(\varphi)$$

$$\varphi \;::=\; \Phi_1 \,\mathsf{U}\, \Phi_2 \mid \Phi_1 \,\mathsf{U}^{\leqslant n}\, \Phi_2$$

- $s_0 s_1 s_2 \ldots \models \Phi \,\mathsf{U}^{\leqslant n}\, \Psi$ if Φ holds until Ψ holds within n steps
- $s \models \mathbb{P}_J(\varphi)$ if probability that paths starting in s fulfill φ lies in J

abbreviate $\mathbb{P}_{[0,0.5]}(\varphi)$ by $\mathbb{P}_{\leqslant 0.5}(\varphi)$ and $\mathbb{P}_{]0,1]}(\varphi)$ by $\mathbb{P}_{>0}(\varphi)$

Derived operators

$$\Diamond \Phi \;=\; \mathsf{true}\,\mathsf{U}\,\Phi$$

$$\Diamond^{\leqslant n} \Phi \;=\; \mathsf{true}\,\mathsf{U}^{\leqslant n}\,\Phi$$

$$\mathbb{P}_{\leqslant p}\big(\Box \Phi\big) \;=\; \mathbb{P}_{\geqslant 1-p}\big(\Diamond \neg \Phi\big)$$

$$\mathbb{P}_{]p,q]}\big(\Box^{\leqslant n} \Phi\big) \;=\; \mathbb{P}_{[1-q,1-p[}\big(\Diamond^{\leqslant n} \neg \Phi\big)$$

operators like weak until W or release R can be derived analogously

Example properties

- Transient probabilities: $\mathbb{P}_{\geqslant 0.92}\left(\Diamond^{=137}\ \mathit{goal}\right)$
- With probability $\geqslant 0.92$, a goal state is reached legally:

$$\mathbb{P}_{\geqslant 0.92}\left(\neg\ \mathit{illegal}\ \mathsf{U}\ \mathit{goal}\right)$$

- . . . in maximally 137 steps: $\mathbb{P}_{\geqslant 0.92}\left(\neg\ \mathit{illegal}\ \mathsf{U}^{\leqslant 137}\ \mathit{goal}\right)$
- . . . once there, remain there almost always for the next 31 jumps:

$$\mathbb{P}_{\geqslant 0.92}\left(\neg\ \mathit{illegal}\ \mathsf{U}^{\leqslant 137}\ \mathbb{P}_{=1}(\Box^{[0,31]}\ \mathit{goal})\right)$$

PCTL semantics (1)

$\mathcal{D}, s \models \Phi$ if and only if formula Φ holds in state s of DTMC $\mathcal{D}$

Relation $\models$ is defined by:

$$
\begin{array}{lll}
s \models a & \text{iff} & a \in L(s) \\
s \models \neg\, \Phi & \text{iff} & \text{not } (s \models \Phi) \\
s \models \Phi \ \vee \ \Psi & \text{iff} & (s \models \Phi) \text{ or } (s \models \Psi) \\
s \models \mathbb{P}_{J}(\varphi) & \text{iff} & \Pr(s \models \varphi) \in J
\end{array}
$$

where $\Pr(s \models \varphi) \ = \ \Pr_s\{\pi \in \mathit{Paths}(s) \mid \pi \models \varphi\}$

PCTL semantics (2)

A *path* in $\mathcal{D}$ is an infinite sequence $s_0\, s_1\, s_2 \ldots$ with $\mathbf{P}(s_i, s_{i+1}) > 0$

Semantics of path-formulas is defined as in CTL:

$$\pi \models \Phi \,\mathsf{U}\, \Psi \quad \text{iff} \quad \exists n \geqslant 0.(\, s_n \models \Psi \,\wedge \forall 0 \leqslant i < n.\, s_i \models \Phi\,)$$

$$\pi \models \Phi \,\mathsf{U}^{\leqslant n}\, \Psi \quad \text{iff} \quad \exists k \geqslant 0.(\, k \leqslant n \,\wedge\, s_k \models \Psi \,\wedge\, \forall 0 \leqslant i < k.\, s_i \models \Phi\,)$$

For any PCTL path formula φ and state s of DTMC $\mathcal{D}$
the set $\{\pi \in \mathit{Paths}(s) \mid \pi \models \varphi\}$ is measurable

PCTL model checking

- Check whether state s in a DTMC satisfies a PCTL formula:
  - compute recursively the set $Sat(\Phi)$ of states that satisfy Φ
  - check whether state s belongs to $Sat(\Phi)$

  $\Rightarrow$ bottom-up traversal of the parse tree of Φ (like for CTL)

- For the propositional fragment: as for CTL

- How to compute $Sat(\Phi)$ for the probabilistic operators?

Checking probabilistic reachability

- $s \models \mathbb{P}_J(\Phi \,\mathsf{U}^{\leqslant h}\, \Psi)$ if and only if $Prob(s, \Phi \,\mathsf{U}^{\leqslant h}\, \Psi) \in J$
- $Prob(s, \Phi \,\mathsf{U}^{\leqslant h}\, \Psi)$ is the least solution of: (Hansson & Jonsson, 1990)
  - 1 if $s \models \Psi$
  - for $h > 0$ and $s \models \Phi \wedge \neg\Psi$:

$$\sum_{s' \in S} \mathbf{P}(s, s') \cdot Prob(s', \Phi \,\mathsf{U}^{\leqslant h-1}\, \Psi)$$

  - 0 otherwise
- Standard reachability for $\mathbb{P}_{>0}(\Phi \,\mathsf{U}^{\leqslant h}\, \Psi)$ and $\mathbb{P}_{\geqslant 1}(\Phi \,\mathsf{U}^{\leqslant h}\, \Psi)$
  - for efficiency reasons (avoiding solving system of linear equations)

Reduction to transient analysis

- Make all Ψ- and all $\neg\,(\Phi \vee \Psi)$-states absorbing in $\mathcal{D}$
- Check $\Diamond^{=h}\,\Psi$ in the obtained DTMC $\mathcal{D}'$
- This is a standard transient analysis in $\mathcal{D}'$:

$$\sum_{s' \models \Psi} \Pr_{s}\{\pi \in \mathit{Paths}(s) \mid \sigma[h] = s'\}$$

  - compute by $(\mathbf{P}')^h \cdot \iota_\Psi$ where ι_Ψ is the characteristic vector of $Sat(\Psi)$

$\Rightarrow$ Matrix-vector multiplication

Summary of model-checking DTMCs

(Hansson & Jonsson, 1990)

- Recursive descent over parse tree of Φ
- Probabilistic (bounded) reachability:
  - graph analysis to obtain states with positive probability to reach Ψ
  - solving a linear equation system, or matrix-vector multiplication
- Worst case time-complexity:

$$\mathcal{O}(|\Phi| \cdot k_{max} \cdot |\, S \,| \cdot |\, \mathbf{P} \,|)$$

$|\Phi|$ the length of Φ and k_{max} the largest bound ($\neq \infty$) of an until in Φ

- Tools: TPWB, ProbVerus, E $\vdash$ MC^2 , Prism, MRMC, APMC*

A synchronous leader election protocol

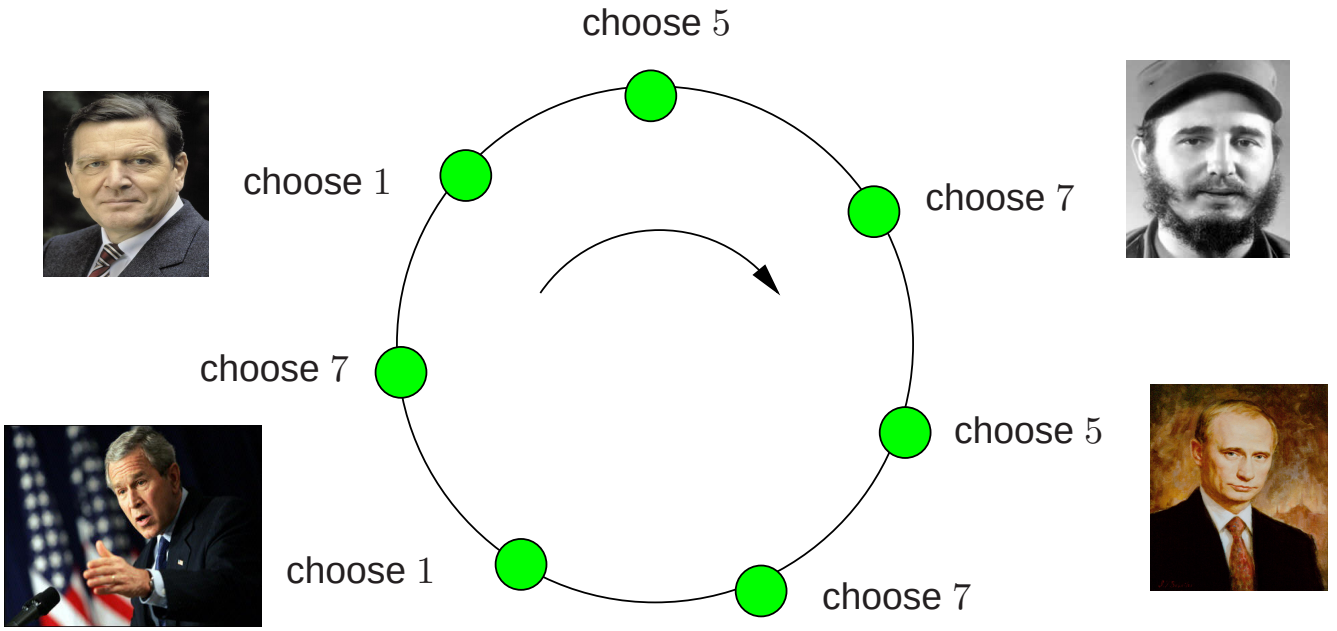
- A round-based protocol in a synchronous ring of $N > 2$ nodes
  - the nodes proceed in a lock-step fashion
  - each slot = 1 message is read + 1 state change + 1 message is sent

  $\Rightarrow$ this synchronous computation yields a DTMC!

- Each round starts by each node choosing a uniform id $\in \{ 1, \ldots, K \}$
- Nodes pass their selected id around the ring
- If there is a unique id, the node with the maximum unique id is leader
- If not, start another round and try again . . .

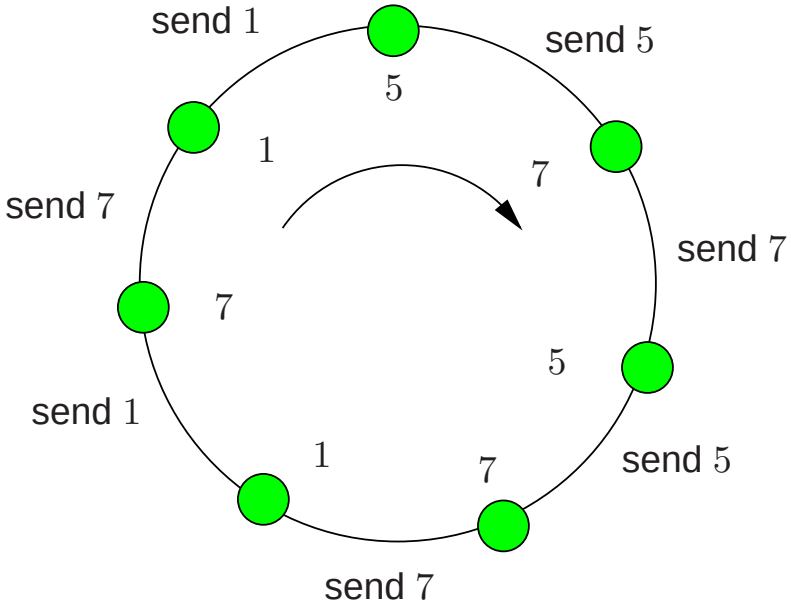
(Itai & Rodeh, 1990) © PRISM web-page

Leader election



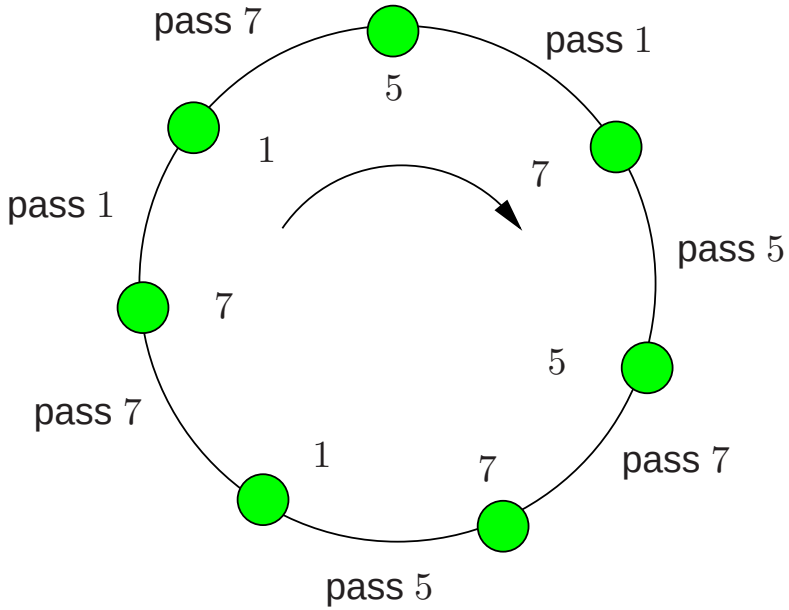
probabilistically choose an id from $[1 \ldots K]$

Leader election



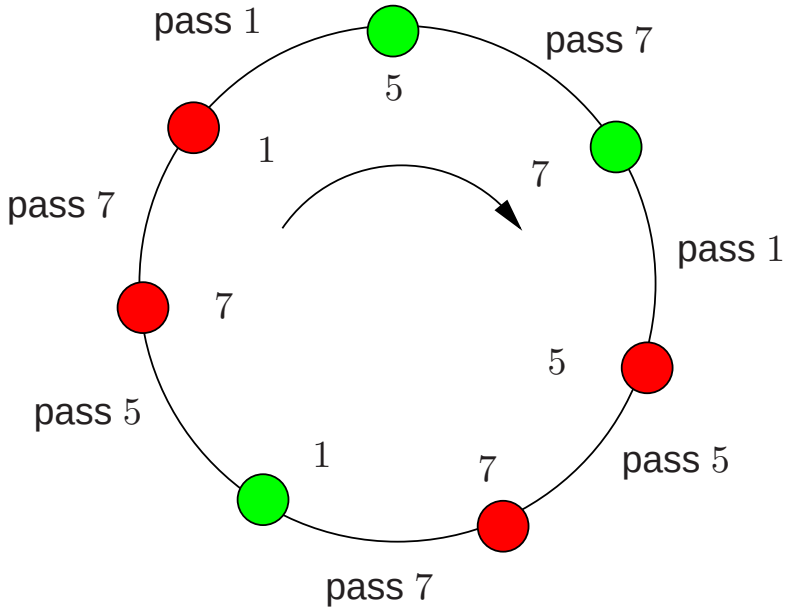
send your selected id to your neighbour

Leader election



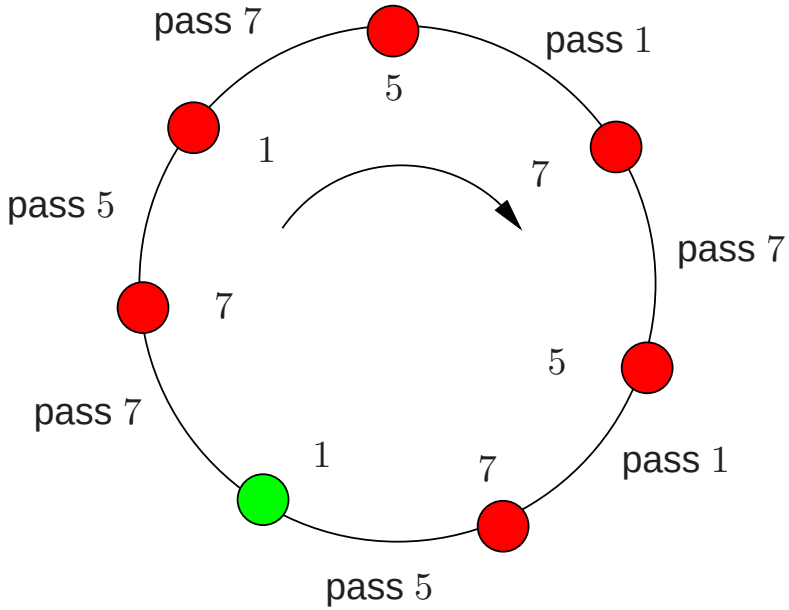
pass the received id, and check uniqueness own id

Leader election



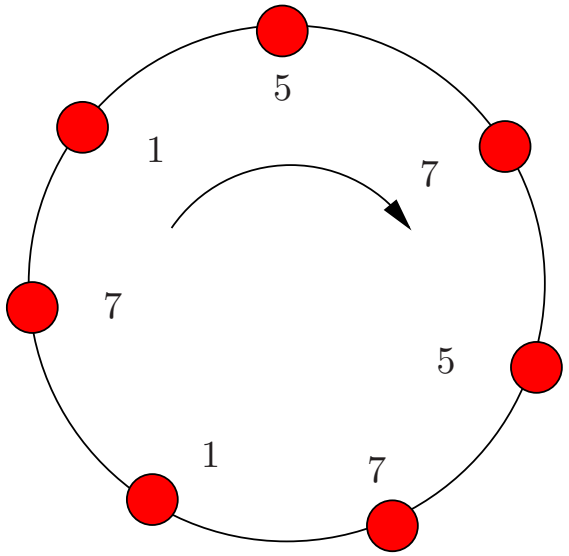
pass the received id, and check uniqueness own id

Leader election



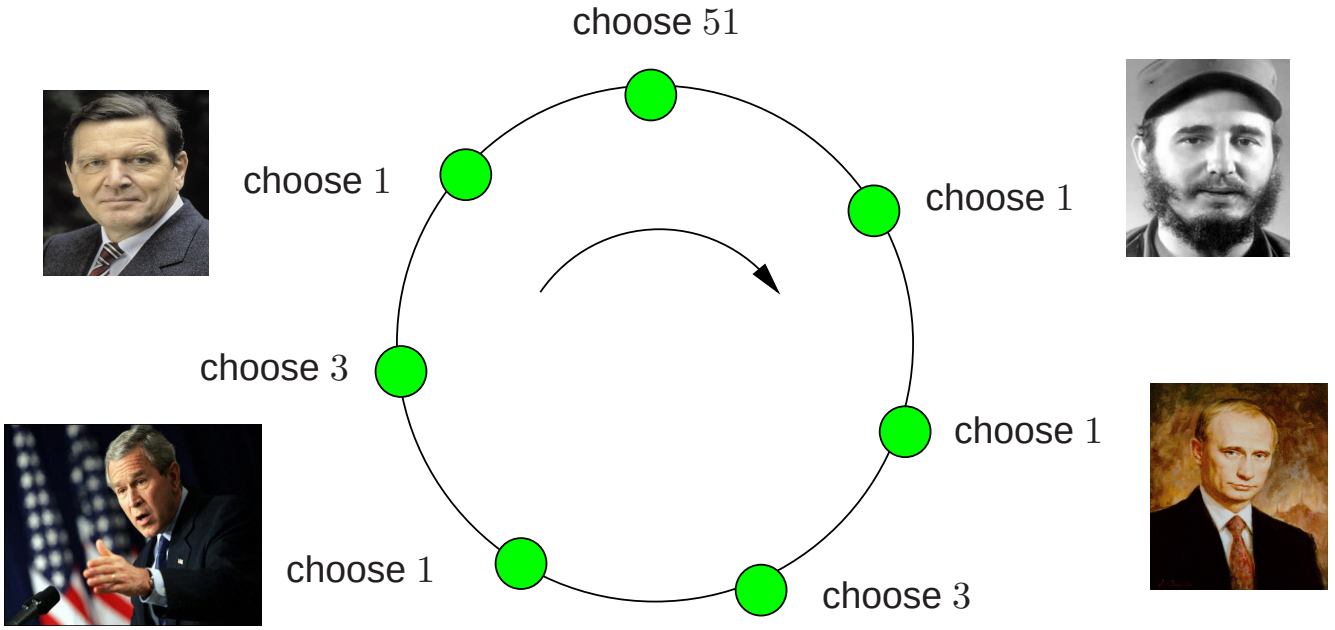
pass the received id, and check uniqueness own id

End of 1st round



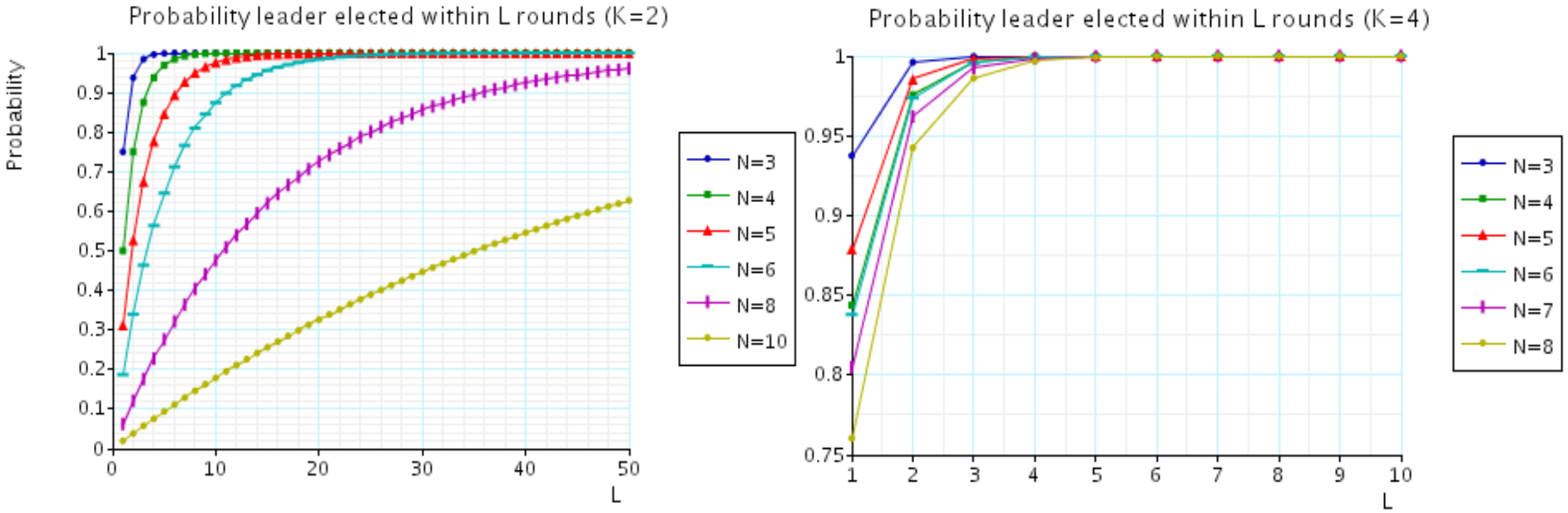
no unique leader has been elected

Start a new round



new round and new chances!

Probability to elect a leader within L rounds



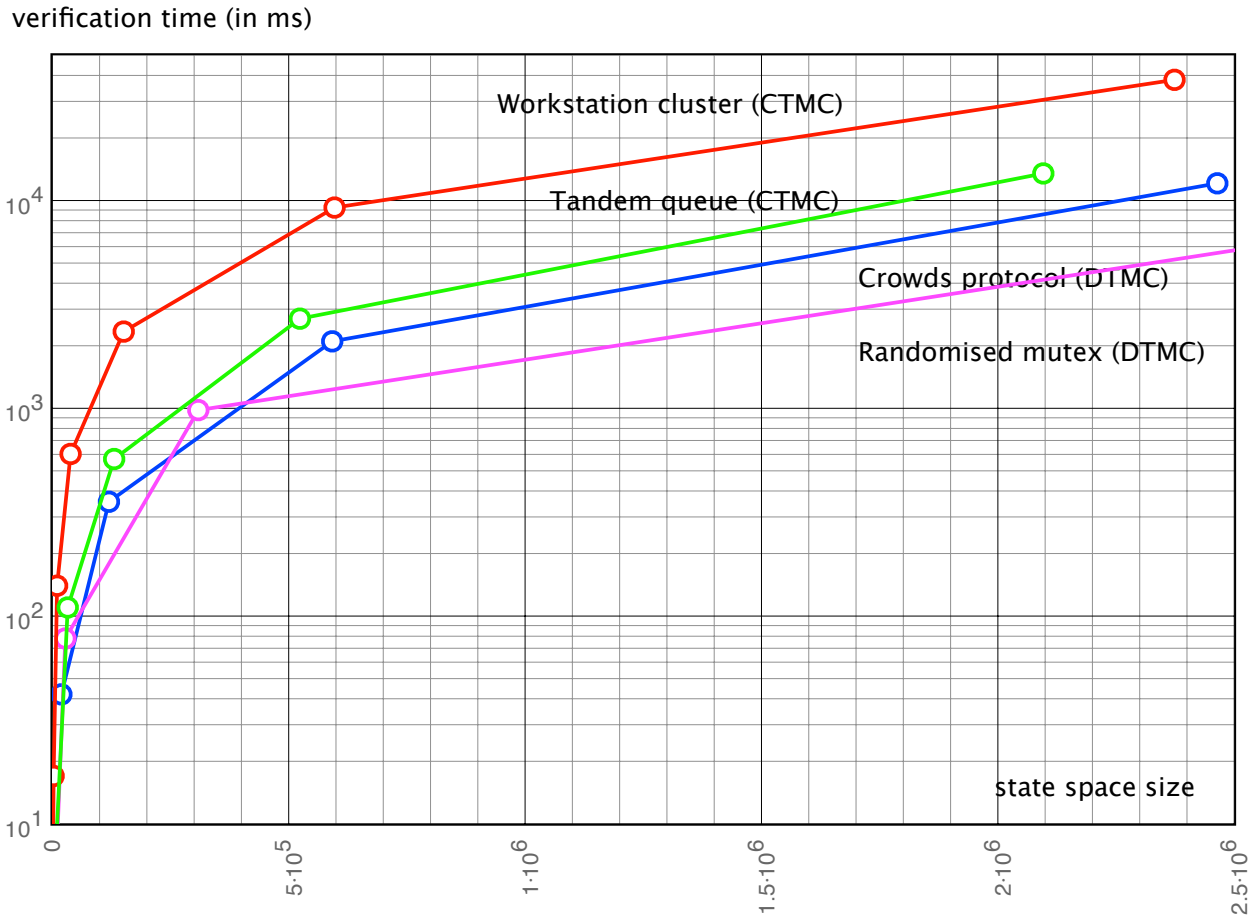
$\mathbb{P}_{\leqslant q}(\Diamond^{\leqslant (N+1)\cdot L}$ *leader elected*$)$ © PRISM web-page

Markov reward model checker (MRMC)

(Zapreev & Meyer-Kayser, 2000/2005)

- Supports DTMCs, CTMCs and cost-based extensions thereof
  - temporal logics: P(R)CTL and CS(R)L
  - bounded until, long run properties, and interval bounded until
- Sparse-matrix representation
- Command-line tool (in C)
  - experimental platform for new (e.g., reward) techniques
  - back-end of GreatSPN, PEPA WB, PRISM and stochastic GG tool
  - freely downloadable under Gnu GPL license
- Experiments: Pentium 4, 2.66 GHz, 1 GB RAM

Verification times



Counterexamples

- Counterexamples are of utmost importance:
  - diagnostic feedback, the key to abstraction-refinement, schedule synthesis . . .
- LTL: counterexamples are finite paths
  - $\Box\Phi$: a path leading to a $\neg\Phi$-state
  - $\Diamond\Phi$: a $\neg\Phi$-path leading to a $\neg\Phi$ cycle
  - BFS yields shortest counterexamples
- Universal CTL$\setminus$LTL: trees or proof-like counterexample
- Existential CTL: witnesses, annotated counterexample
- Probabilistic CTL/LTL:
  - what is a counterexample?, how to determine it?, smallest?

Evidences

$$s \not\models \mathbb{P}_{\leqslant p}(\varphi)$$

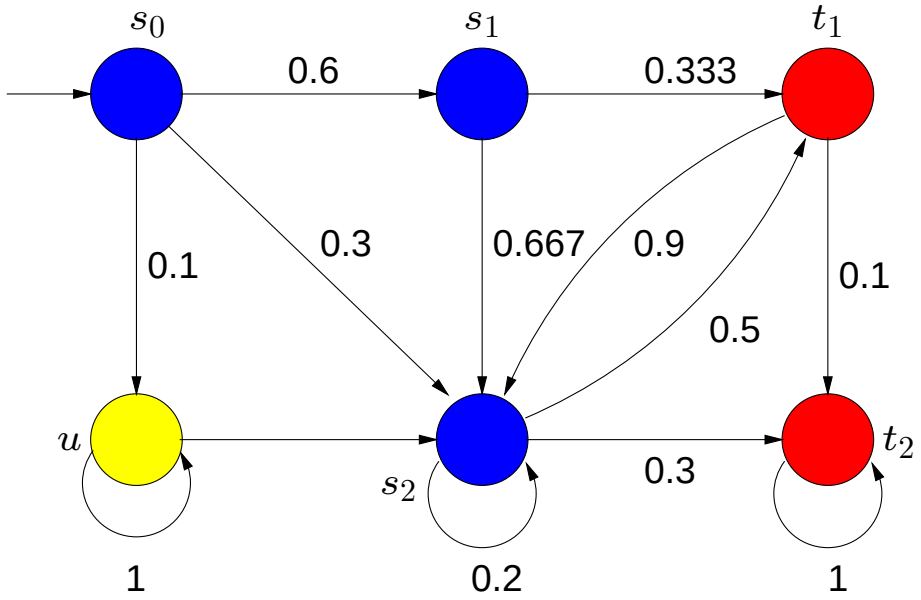
$$\Leftrightarrow \text{ not } \Pr\{\, \sigma \in \mathit{Paths}(s) \mid \sigma \models \varphi \,\} \leqslant p$$

$$\Leftrightarrow \Pr\{\, \sigma \in \mathit{Paths}(s) \mid \sigma \models \varphi \,\} > p$$

- An *evidence* is a finite path starting in s satisfying φ
- A *strongest* evidence is an evidence σ such that:

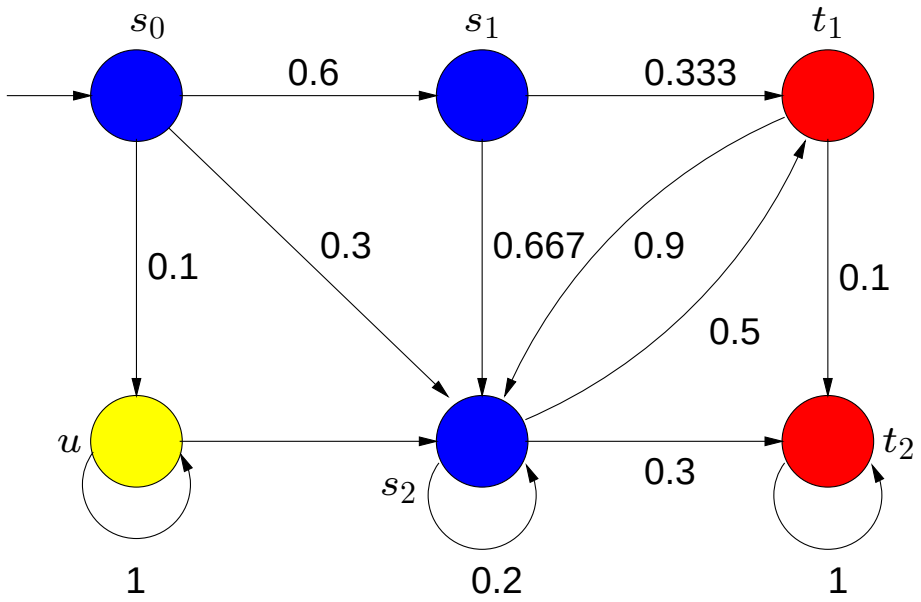
  $\Pr\{\, \sigma \,\} \geqslant \Pr\{\, \widehat{\sigma} \,\}$ for any evidence $\widehat{\sigma}$

Evidences for $s_0 \not\models \mathbb{P}_{\leqslant \frac{1}{2}}(a \,\mathsf{U}\, b)$



evidences	prob.
$\sigma_1 = s_0\, s_1\, t_1$	0.2
$\sigma_2 = s_0\, s_1\, s_2\, t_1$	0.2
$\sigma_3 = s_0\, s_2\, t_1$	0.15
$\sigma_4 = s_0\, s_1\, s_2\, t_2$	0.12
$\sigma_5 = s_0\, s_2\, t_2$	0.09
$\ldots$	$\ldots$

Strongest evidences



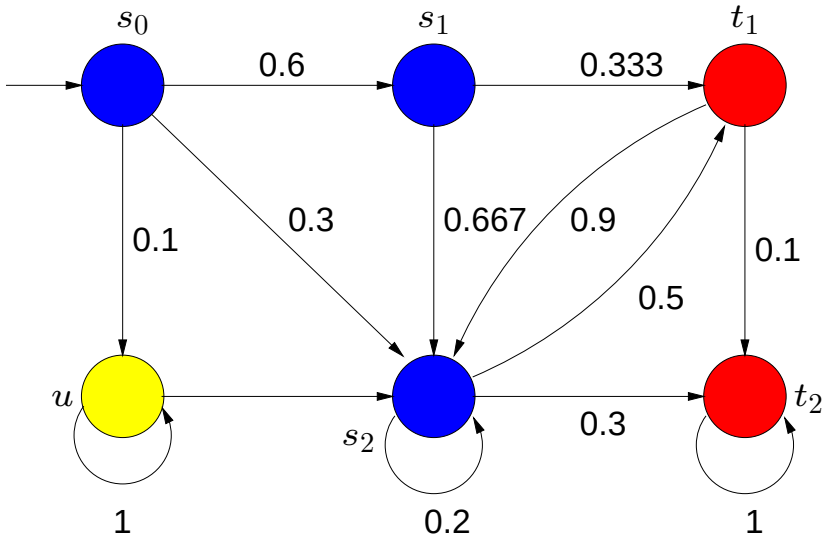
evidences	prob.
$\sigma_1 = s_0\, s_1\, t_1$	0.2
$\sigma_2 = s_0\, s_1\, s_2\, t_1$	0.2
$\sigma_3 = s_0\, s_2\, t_1$	0.15
$\sigma_4 = s_0\, s_1\, s_2\, t_2$	0.12
$\sigma_5 = s_0\, s_2\, t_2$	0.09
$\ldots$	$\ldots$

Counterexamples

For $s \not\models \mathbb{P}_{\leqslant p}(\varphi)$:

- A *counterexample* is a set C of evidences such that $\Pr\{\, C \,\} > p$
  - counterexamples are always *finite* sets (of finite paths)
- A *minimal* counterexample is a counterexample C such that:
  - $|C| \leqslant |C'|$ for any counterexample C'
- A counterexample C is *smallest, most indicative* whenever:
  - C is minimal, and $\Pr\{\, C \,\} \geqslant \Pr\{\, C' \,\}$ for any minimal counterexample C'

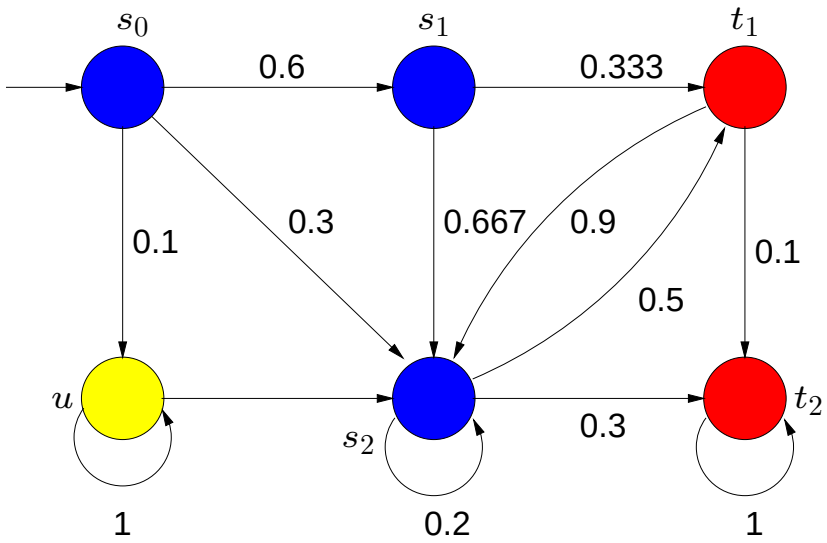
Counterexamples for $s_0 \not\models \mathbb{P}_{\leqslant \frac{1}{2}}(a \,\mathsf{U}\, b)$



evidences	prob.
$\sigma_1 = s_0\, s_1\, t_1$	0.2
$\sigma_2 = s_0\, s_1\, s_2\, t_1$	0.2
$\sigma_3 = s_0\, s_2\, t_1$	0.15
$\sigma_4 = s_0\, s_1\, s_2\, t_2$	0.12
$\sigma_5 = s_0\, s_2\, t_2$	0.09

counterexample	card.	prob.
$\{\, \sigma_1, \ldots, \sigma_5 \,\}$	5	0.76
$\{\, \sigma_1$ or $\sigma_2, \ldots, \sigma_5 \,\}$	4	0.56
$\{\, \sigma_1, \sigma_2, \sigma_4 \,\}$	3	0.52
$\{\, \sigma_1, \sigma_2, \sigma_3 \,\}$	3	0.55

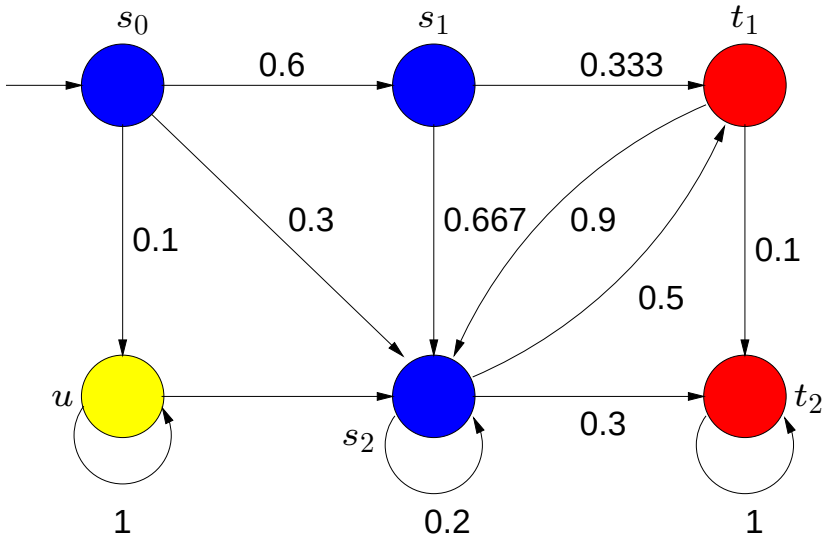
Counterexamples for $s_0 \not\models \mathbb{P}_{\leqslant \frac{1}{2}}(a \,\mathsf{U}\, b)$



evidences	prob.
$\sigma_1 = s_0\, s_1\, t_1$	0.2
$\sigma_2 = s_0\, s_1\, s_2\, t_1$	0.2
$\sigma_3 = s_0\, s_2\, t_1$	0.15
$\sigma_4 = s_0\, s_1\, s_2\, t_2$	0.12
$\sigma_5 = s_0\, s_2\, t_2$	0.09

counterexample	card.	prob.
$\{\, \sigma_1, \ldots, \sigma_5 \,\}$	5	0.76
$\{\, \sigma_1$ or $\sigma_2, \ldots, \sigma_5 \,\}$	4	0.56
minimal $\longrightarrow \{\, \sigma_1, \sigma_2, \sigma_4 \,\}$	3	0.52
minimal $\longrightarrow \{\, \sigma_1, \sigma_2, \sigma_3 \,\}$	3	0.55

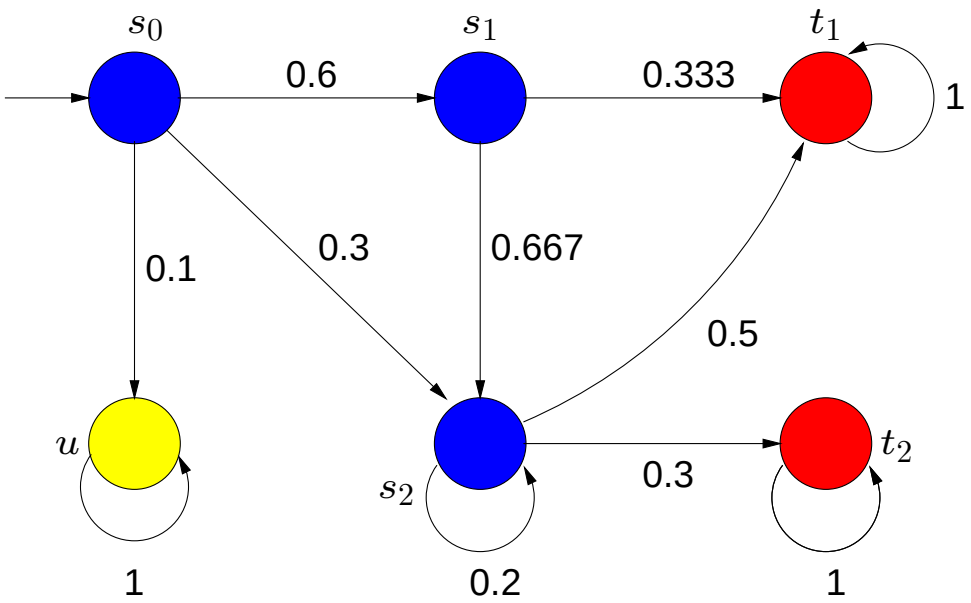
Counterexamples for $s_0 \not\models \mathbb{P}_{\leqslant \frac{1}{2}}(a \,\mathsf{U}\, b)$



evidences	prob.
$\sigma_1 = s_0\, s_1\, t_1$	0.2
$\sigma_2 = s_0\, s_1\, s_2\, t_1$	0.2
$\sigma_3 = s_0\, s_2\, t_1$	0.15
$\sigma_4 = s_0\, s_1\, s_2\, t_2$	0.12
$\sigma_5 = s_0\, s_2\, t_2$	0.09

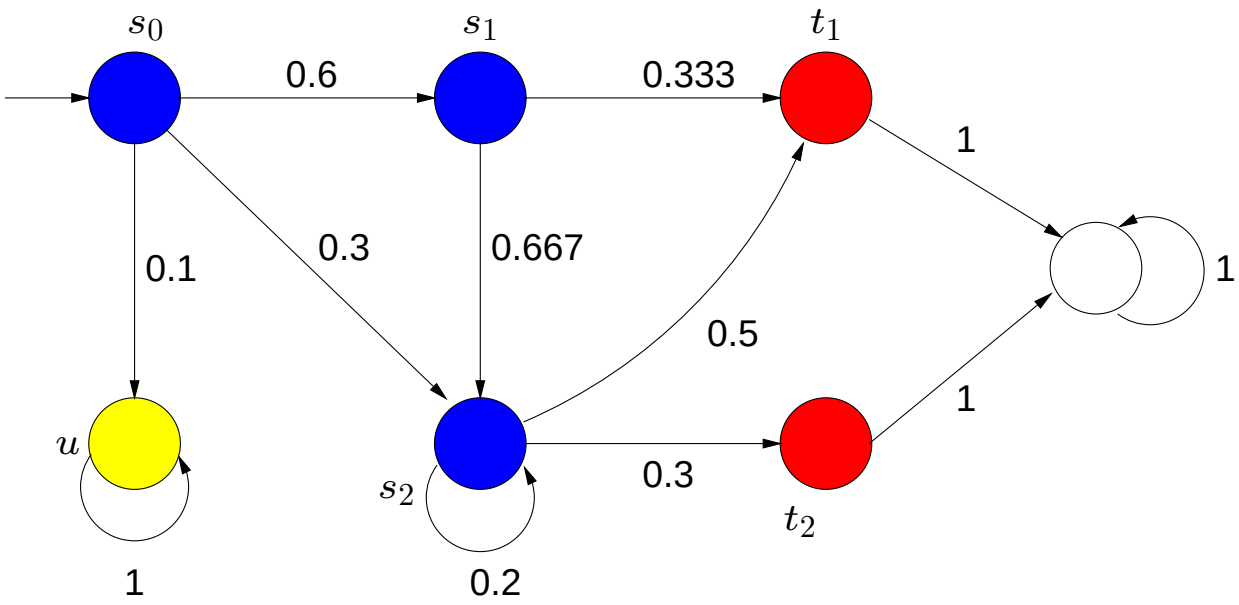
counterexample	card.	prob.
$\{\, \sigma_1, \ldots, \sigma_5 \,\}$	5	0.76
$\{\, \sigma_1$ or $\sigma_2, \ldots, \sigma_5 \,\}$	4	0.56
$\{\, \sigma_1, \sigma_2, \sigma_4 \,\}$	3	0.52
smallest $\longrightarrow$ $\{\, \sigma_1, \sigma_2, \sigma_3 \,\}$	3	0.55

Adapting the Markov chain



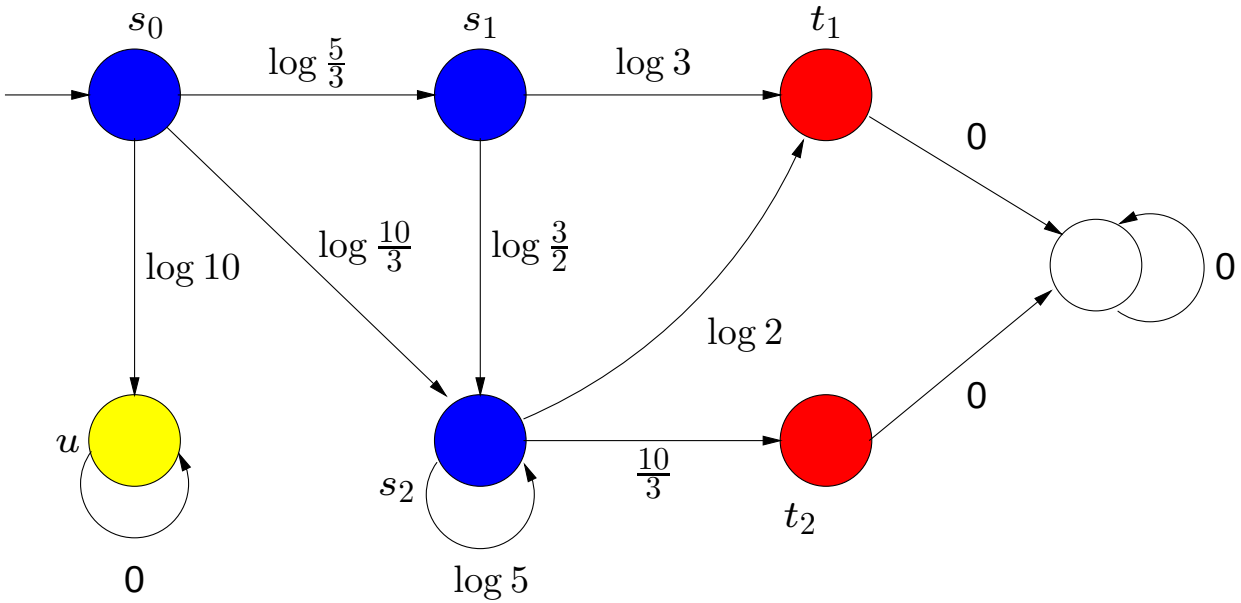
Step 1: make all Ψ-states and all $\neg\Phi \wedge \neg\Psi$-states absorbing

Adapting a bit more



Step 2: insert a sink state and redirect all outgoing edges of Ψ-states to it

A weighted digraph



Step 3: turn it into a weighted digraph with $w(s, s') = \log\left(\mathbf{P}(s, s')^{-1}\right)$

Shortest path problems

For finite path $\sigma = s_0\, s_1\, s_2\, \ldots\, s_n$:

$$
\begin{aligned}
w(\sigma) &= w(s_0, s_1) + w(s_1, s_2) + \ldots + w(s_{n-1}, s_n) \\
&= \log \tfrac{1}{\mathbf{P}(s_0,s_1)} + \log \tfrac{1}{\mathbf{P}(s_1,s_2)} + \ldots + \log \tfrac{1}{\mathbf{P}(s_{n-1},s_n)} \\
&= \log \frac{1}{\mathbf{P}(s_0,s_1)\cdot\mathbf{P}(s_1,s_2)\cdot\ldots\cdot\mathbf{P}(s_{n-1},s_n)} \\
&= \log \tfrac{1}{\Pr\{\,\sigma\,\}}
\end{aligned}
$$

$\underbrace{\Pr\{\,\widehat{\sigma}\,\} \geqslant \Pr\{\,\sigma\,\}}_{\text{in DTMC } \mathcal{D}}$ if and only if $\underbrace{w(\widehat{\sigma}) \leqslant w(\sigma)}_{\text{in digraph } G(\mathcal{D})}$

Bellman equations

- Strongest evidence for unbounded until = shortest path
- for bounded until = hop-bounded shortest path

$\pi_h(s, v)$ is the shortest path (if any) from s to v with at most h hops:

$$\pi_h(s,v) = \begin{cases} s & \text{if } v{=}s \text{ and } h \geqslant 0 \\ \bot & \text{if } v{\neq}s \text{ and } h = 0 \\ \arg\min_u \{ w\left(\pi_{h-1}(s,u) \cdot v\right) \mid (u,v) \in E \} & \text{otherwise} \end{cases}$$

Finding smallest counterexamples

- A counterexample C is smallest if:
  - C is minimal and $\Pr\{ C \} \geqslant \Pr\{ C' \}$ for any minimal C'
- k shortest paths (KSP) problem: find the k SPs between two vertices

  i.e., { shortest path, 2nd shortest path, . . . , kth shortest path }
- Well-studied problem in algorithmics
  - best known time complexity $\mathcal{O}(M+N \cdot \log N+k)$ (Eppstein, 1998)
  - the recursive enumeration algorithm performs better (Jiménez et. al, 1999)

$\Rightarrow$ The shortest counterexample problem (for $h=\infty$) is a KSP problem

- dynamically determine k: generate C incrementally and halt when $\Pr\{ C \} > p$

Recursive enumeration ($h{=}\infty$)

$\pi^k(s,v)$ is the k-th shortest path (if any) from s to v:

$$\pi^k(s,v) = \begin{cases} s & \text{if } v{=}s \text{ and } k{=}1 \\ \bot & \text{if } v{\neq}s \text{ and } k{>}1 \\ \arg\min\{w(\sigma) \mid \sigma \in Q^k(s,v)\} & \text{otherwise} \end{cases}$$

$Q^k(s,v)$ is a set of candidate paths among which $\pi^k(s,v)$ is chosen:

$$Q^k(s,v) = \begin{cases} \{\, \pi^1(s,u) \cdot v \mid (u,v) \in E \,\} & \text{if } k{=}1, v{\neq}s \text{ or } k{=}2, v{=}s \\ \left(Q^{k-1}(s,v) - \{\, \underbrace{\pi^{k'}(s,u) \cdot v}_{\pi^{k-1}(s,v)} \,\}\right) \cup \{\, \pi^{k'+1}(s,u) \cdot v \,\} & \text{if } k{>}1 \end{cases}$$

Time complexity

counterexample problem	shortest path problem	algorithm	time complexity
SE ($h{=}\infty$)	SP	Dijkstra	$\mathcal{O}(M + N \cdot \log N)$
SE ($h{\neq}\infty$)	HSP	BF / Viterbi	$\mathcal{O}(h \cdot M)$
SC ($h{=}\infty$)	KSP	Eppstein	$\mathcal{O}(M + N \cdot \log N + k)$
SC ($h{\neq}\infty$)	HKSP	adapted REA	$\mathcal{O}\left(h \cdot M + h \cdot k \cdot \log N\right)$

$N = |S|$, $M = \#$ transitions, h = hop count, $k = \#$ shortest paths

including rewards yields a non-trivial instance of the NP-complete RSP problem

Traditional model checking

- Bisimulation: (Fisler & Vardi, 1998)
  - preserves μ-calculus
  - . . . obtains significant state space reductions
  - . . . minimization effort significantly exceeds model checking time
- Advantages:
  - fully automated and efficient abstraction technique
  - may be tailored to properties-of-interest
  - enables compositional minimisation
- Does bisimulation in probabilistic model checking pay off?

Probabilistic bisimulation

- Let $\mathcal{D} = (S, \mathbf{P}, L)$ be a DTMC and R an equivalence relation on S
- R is a *probabilistic bisimulation* on S if for any $(s, s') \in R$:

$$L(s) = L(s') \text{ and } \mathbf{P}(s, C) = \mathbf{P}(s', C) \quad \text{for all} \quad C \text{ in } S/R$$

where $\mathbf{P}(s, C) = \sum_{s' \in C} \mathbf{P}(s, s')$ (Larsen & Shou, 1989)

- $s \sim s'$ if $\exists$ a probabilistic bisimulation R on S with $(s, s') \in R$

$$s \sim s' \Leftrightarrow (\forall \Phi \in \textit{PCTL} : s \models \Phi \text{ if and only if } s' \models \Phi)$$

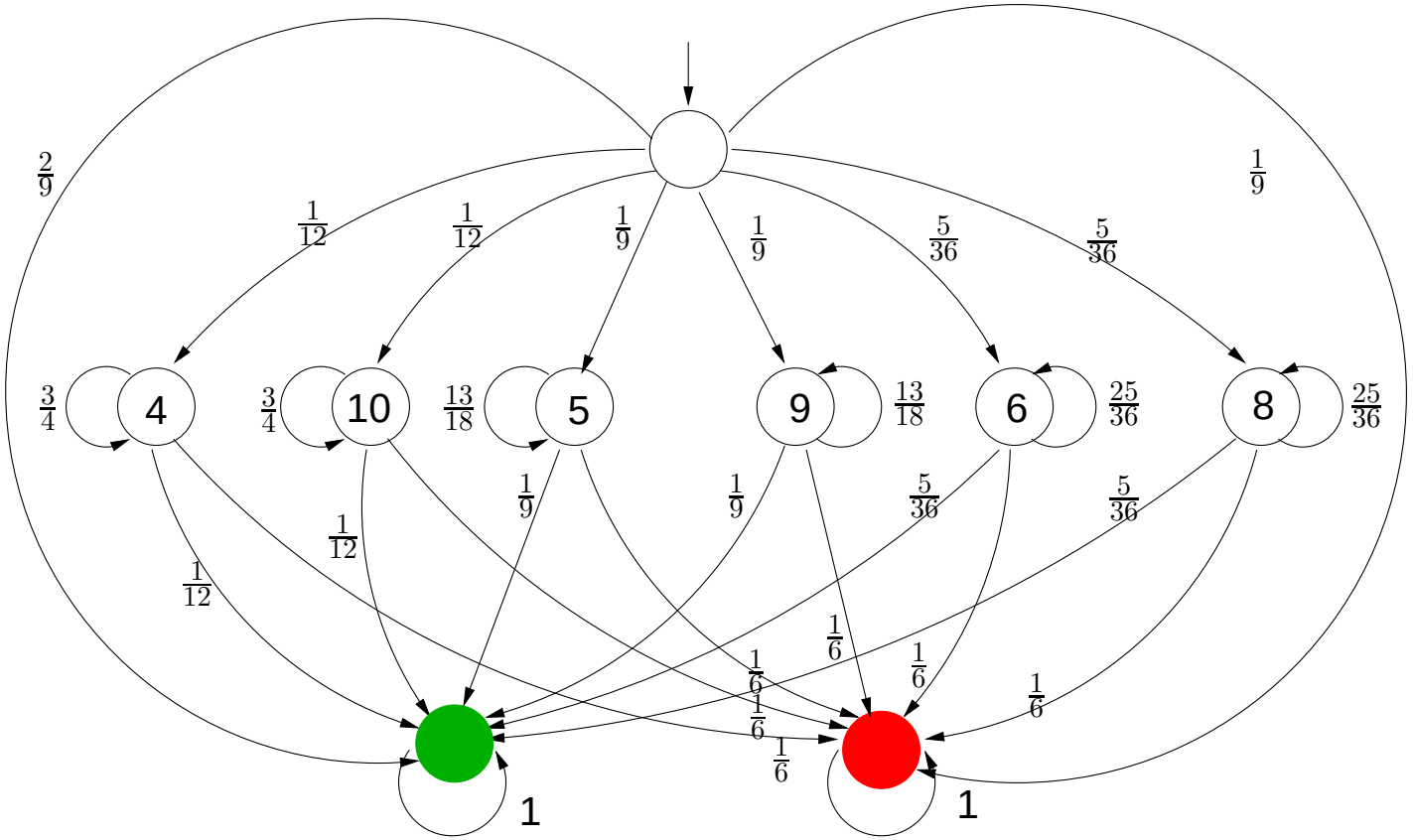
Quotient DTMC under $\sim$

$\mathcal{D}/\!\sim = (S', \mathbf{P}', L')$, the quotient of $\mathcal{D} = (S, \mathbf{P}, L)$ under $\sim$:

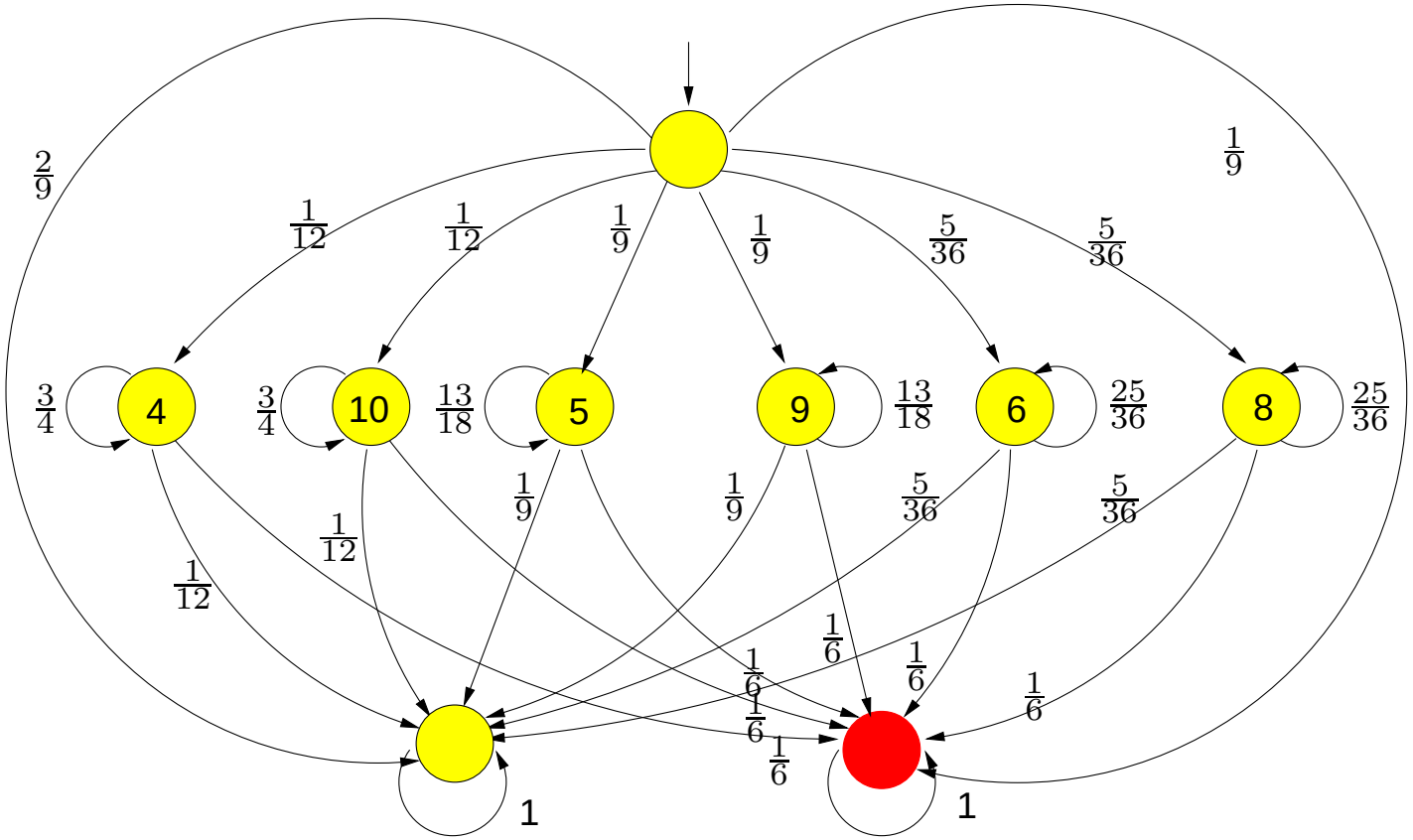
- $S' = S/\!\sim = \{ [s]_\sim \mid s \in S \}$
- $\mathbf{P}'([s]_\sim, C) = \mathbf{P}(s, C)$
- $L'([s]_\sim) = L(s)$

get $\mathcal{D}/\!\sim$ by partition-refinement in time $\mathcal{O}(M \cdot \log N + |AP| \cdot N)$ (Derisavi et al., 2001)

A DTMC model of Craps

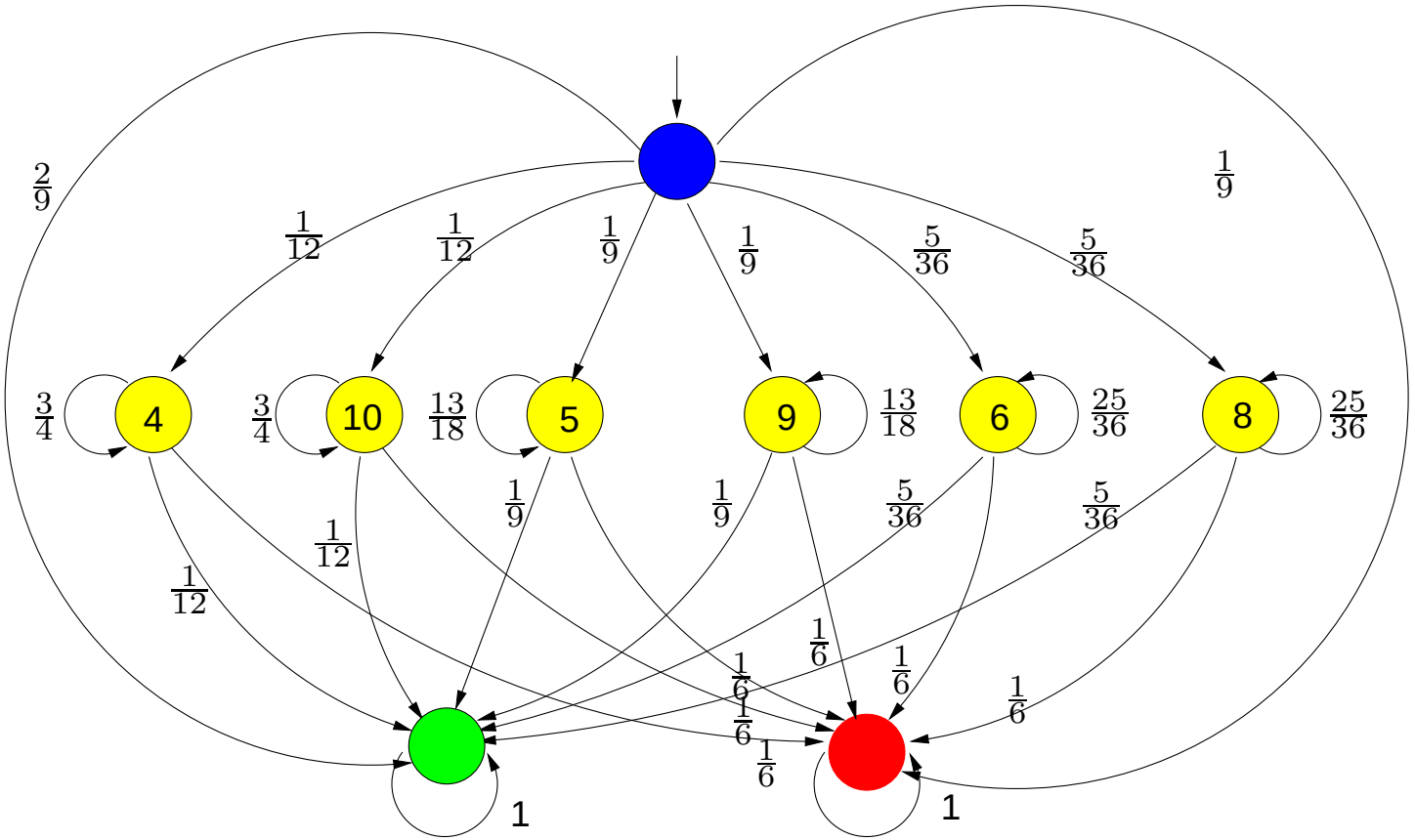


Minimizing Craps



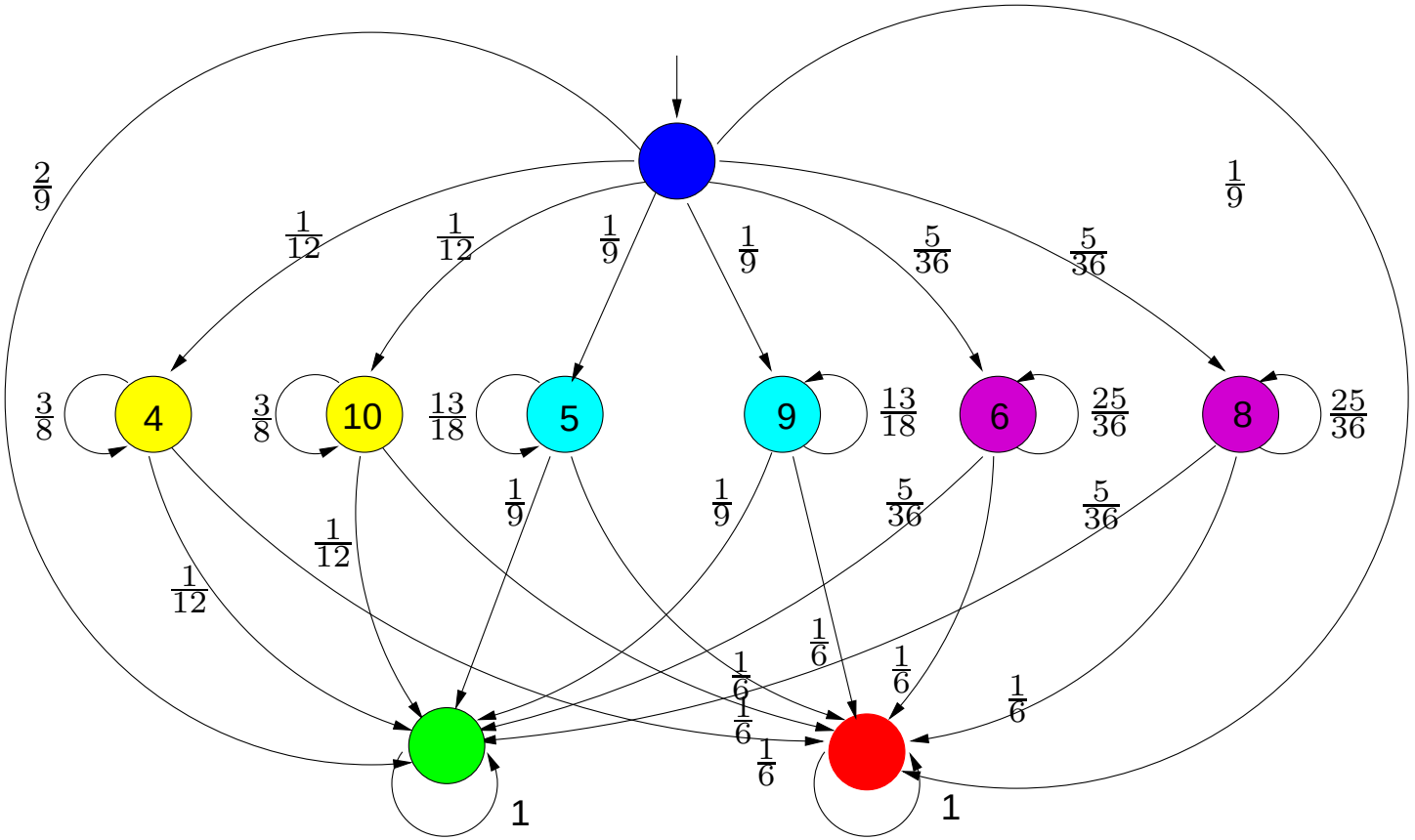
initial partitioning for the atomic propositions $AP = \{ \mathit{loss} \}$

A first refinement



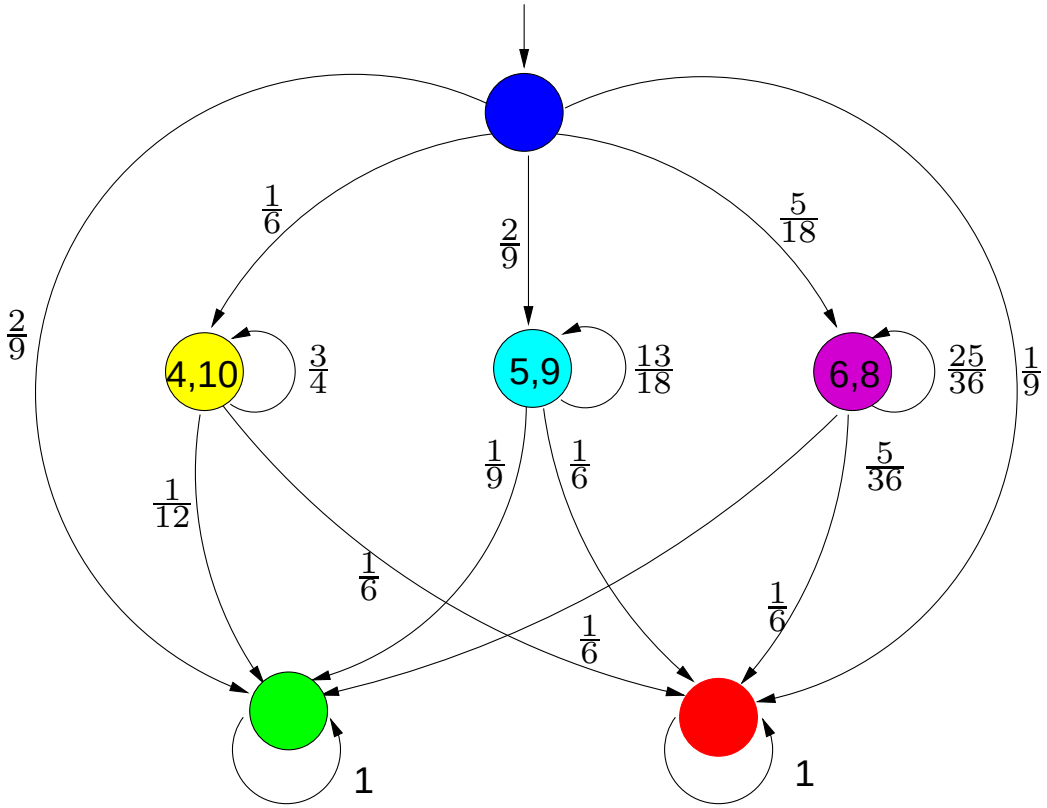
refine (“split”) with respect to the set of red states

A second refinement



refine (“split”) with respect to the set of green states

Quotient DTMC



Property-driven bisimulation

- For DTMC $\mathcal{D}$, set F of PCTL-formulas, and equivalence R on S
- R is a probabilistic F-bisimulation on S if for any $(s, s') \in R$:

$$L_F(s) = L_F(s') \text{ and } \mathbf{P}(s, C) = \mathbf{P}(s', C) \quad \text{for all} \quad C \text{ in } S/R$$

where $L_F(s) = \{ \Phi \in F \mid s \models \Phi \}$ (Baier et al., 2000)

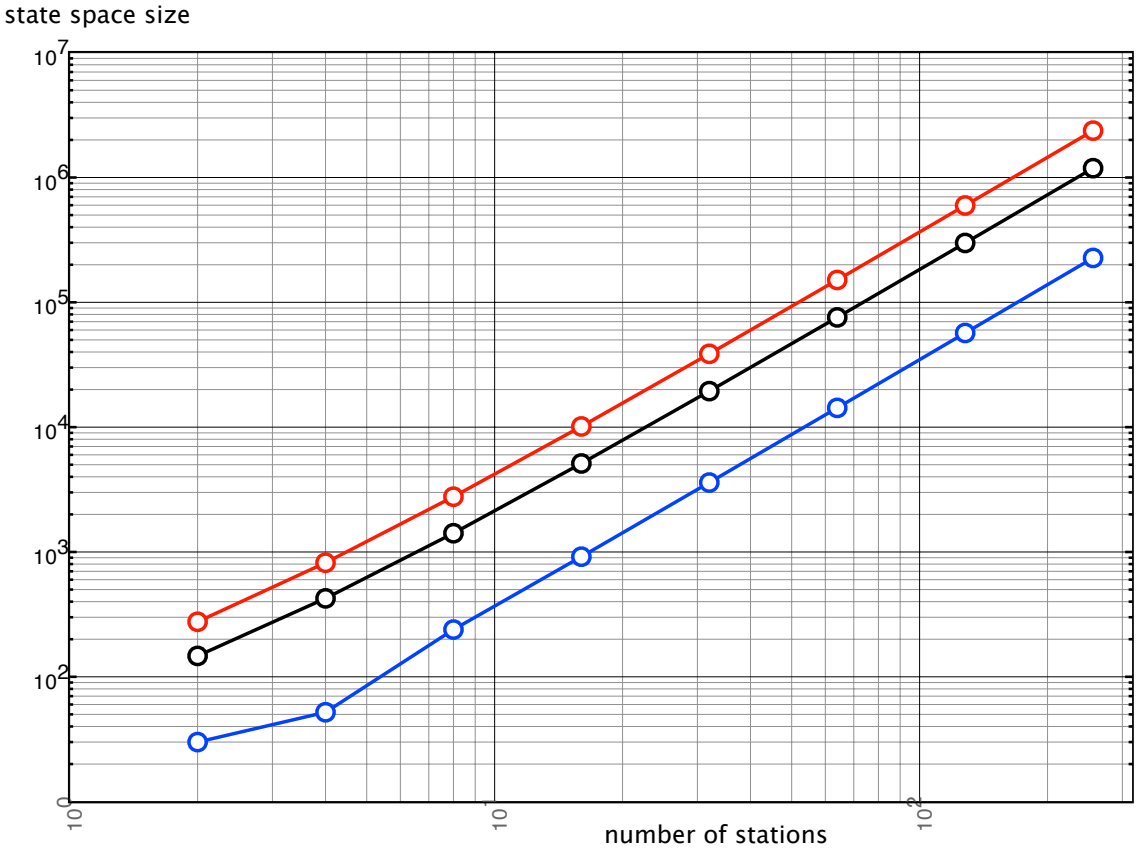
- $s \sim_F s'$ if $\exists$ a probabilistic F-bisimulation R on S with $(s, s') \in R$

$$s \sim_F s' \Leftrightarrow (\forall \Phi \in \mathit{PCTL}_F : s \models \Phi \text{ if and only if } s' \models \Phi)$$

Minimization for Φ until Ψ

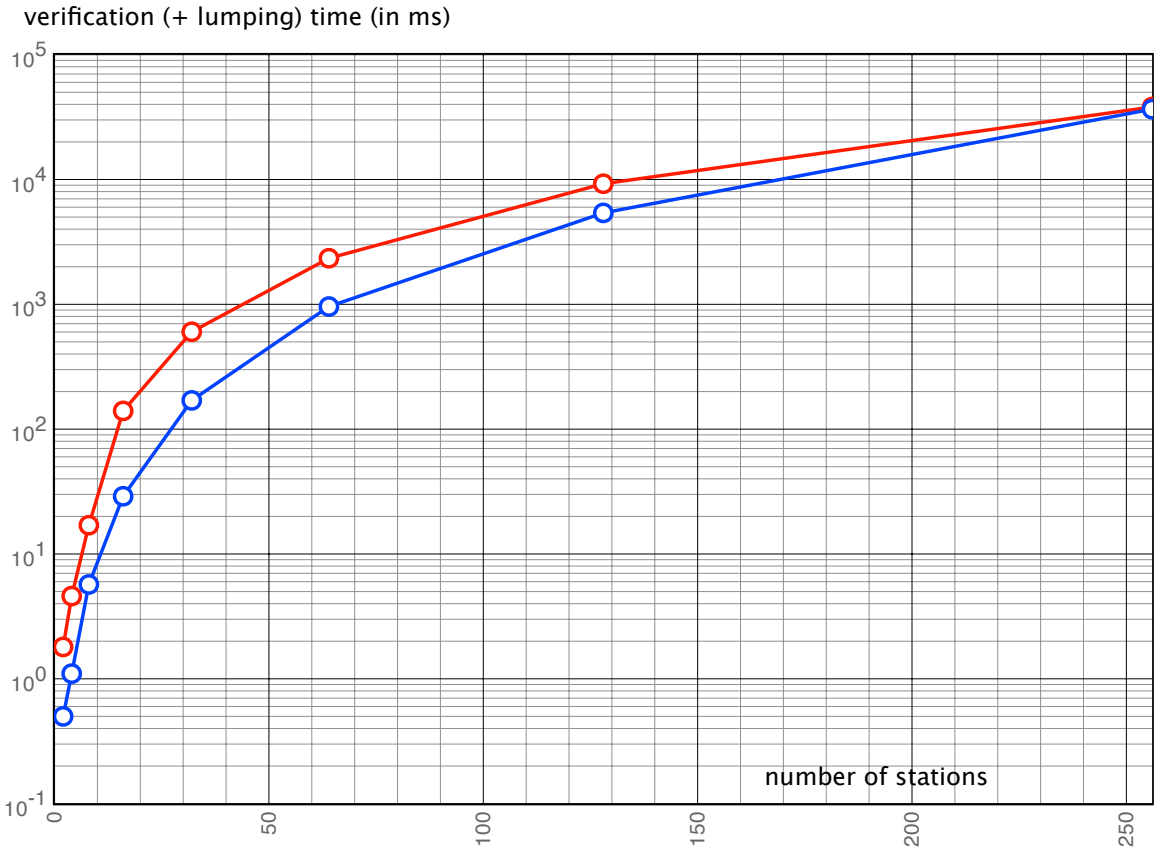
- Initial partition for $\sim$: $s_\Pi = \{\, s' \mid L(s') = L(s) \,\}$
  - independent of the formula to be checked
- Now: exploit the structure of the formula to be checked
- Bounded until:
  - take $F = \{\, \Psi, \neg\Phi \wedge \neg\Psi, \Phi \wedge \neg\Psi \,\}$
  - initial partition $\Pi = \{\, s_\Psi, s_{\neg\Phi\wedge\neg\Psi}, Sat(\Phi \wedge \neg\Psi) \,\}$
  - or, for non-recurrent DTMCs: $\mathbb{P}_{\leqslant 0}(\Phi \;\mathsf{U}\; \Psi)$ instead of $\neg\Phi \wedge \neg\Psi$
- Standard until:
  - take $F = \{\, \underbrace{\mathbb{P}_{\geqslant 1}(\Phi \;\mathsf{U}\; \Psi)}_{\text{single state in } \Pi}, \underbrace{\mathbb{P}_{\leqslant 0}(\Phi \;\mathsf{U}\; \Psi)}_{\text{single state in } \Pi}, \mathbb{P}_{>0}(\Phi \;\mathsf{U}\; \Psi) \wedge \mathbb{P}_{<1}(\Phi \;\mathsf{U}\; \Psi) \,\}$

Workstation cluster (Haverkort et al., 2001)



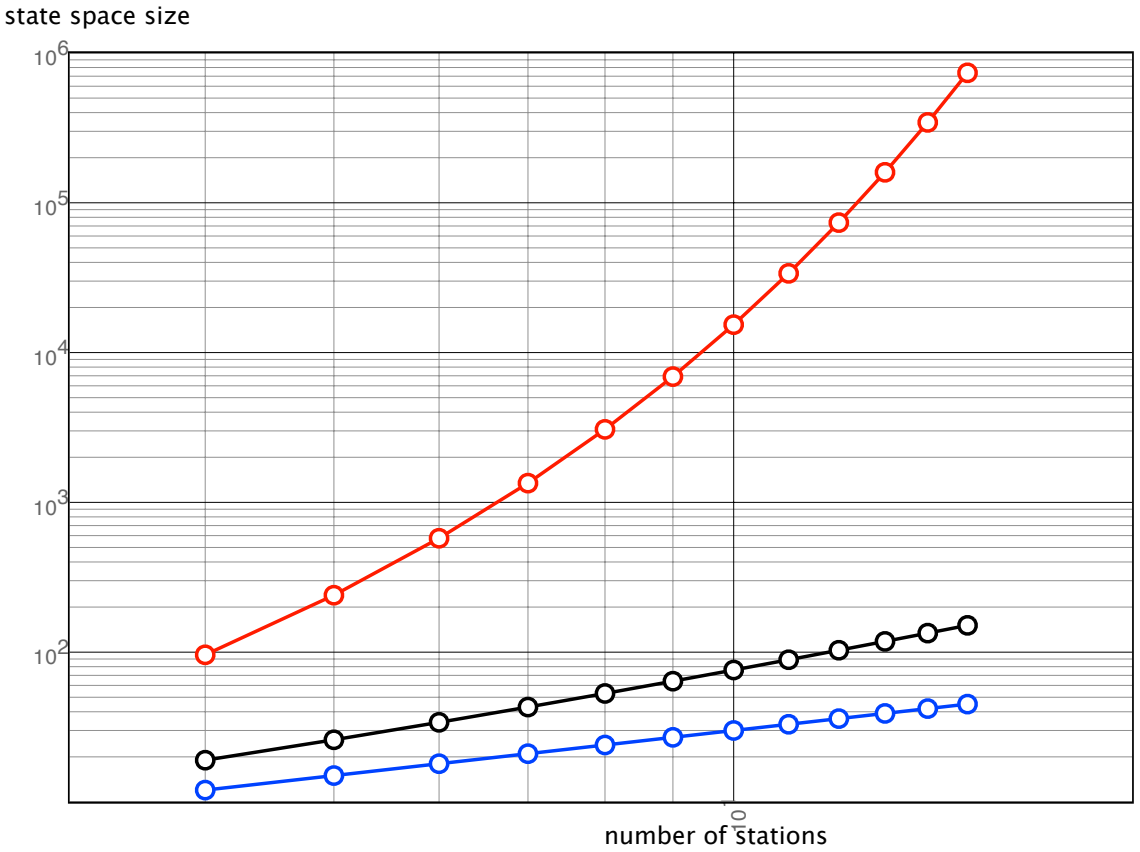
state space reductions for $\mathbb{P}_{\leqslant q}(\mathit{minimum}\ \mathsf{U}^{\leqslant 510}\ \mathit{premium})$

Workstation cluster



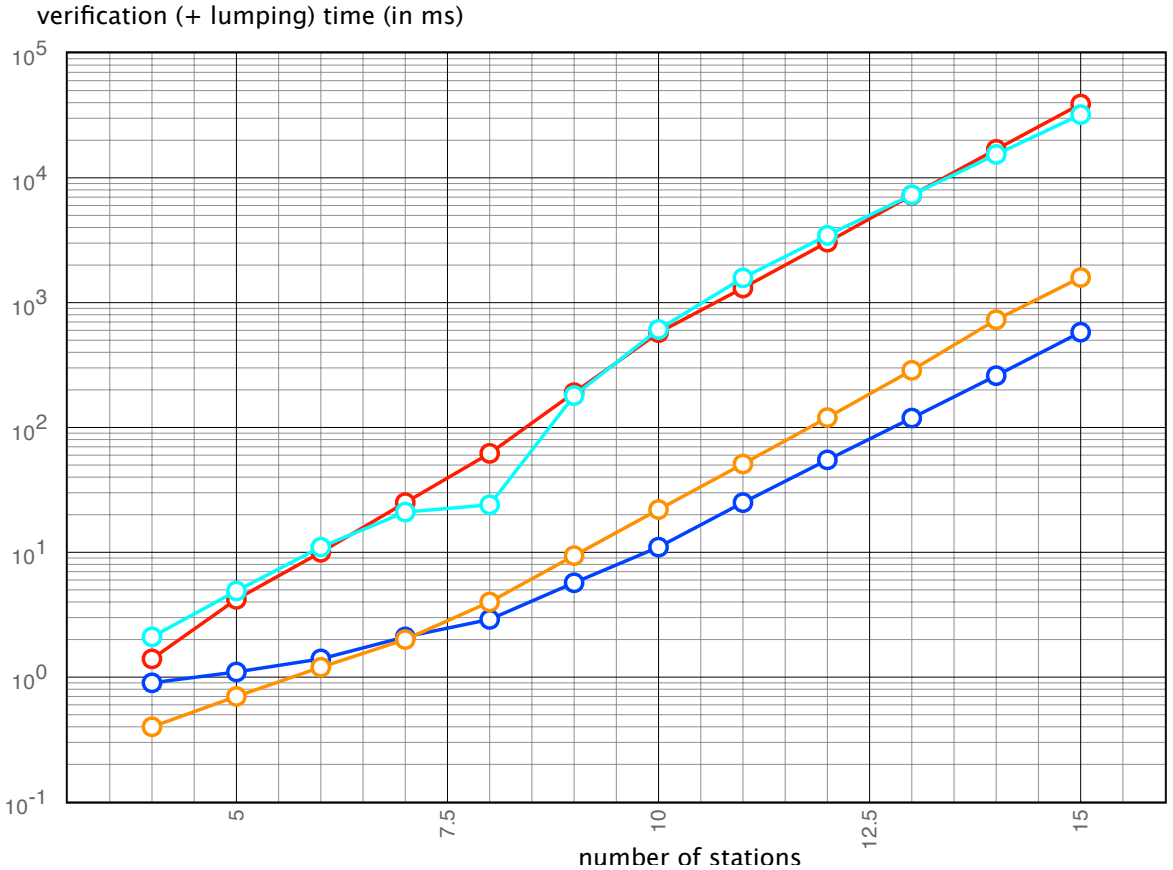
verification (+ lumping) times (in ms) for $\mathbb{P}_{\leqslant q}(\mathit{minimum}\ \mathsf{U}^{\leqslant 510}\ \mathit{premium})$

Cyclic polling system (Ibe & Trivedi, 1989)



state space reductions for $\mathbb{P}_{\leqslant q}(\neg \mathit{serve}_1 \ \mathsf{U}^{\leqslant 1010} \ \mathit{serve}_1)$ and $\mathbb{P}_{\leqslant q}(\neg \mathit{serve}_1 \ \mathsf{U} \ \mathit{serve}_1)$

Cyclic polling system

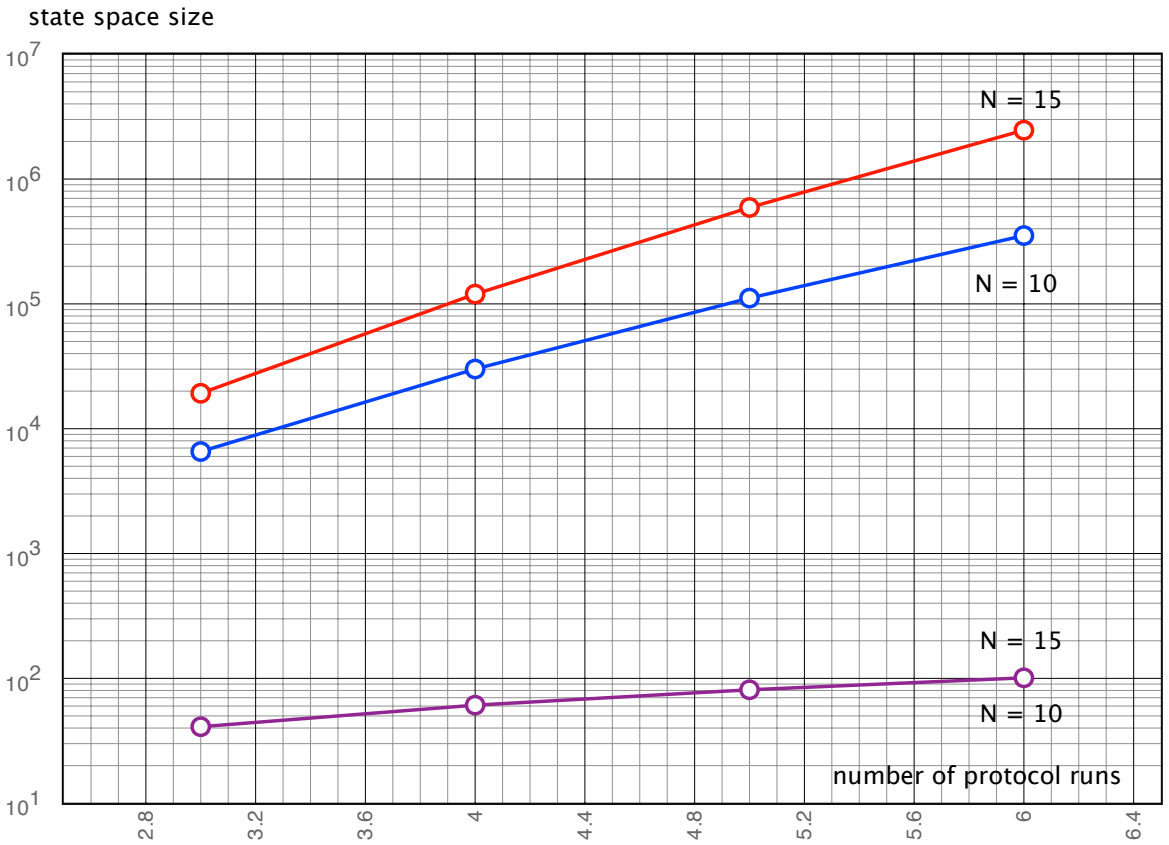


run times for $\mathbb{P}_{\leqslant q}(\neg \mathit{serve}_1 \ \mathsf{U}^{\leqslant 1010} \ \mathit{serve}_1)$ and $\mathbb{P}_{\leqslant q}(\neg \mathit{serve}_1 \ \mathsf{U} \ \mathit{serve}_1)$

Crowds protocol (Reiter & Rubin, 1998)

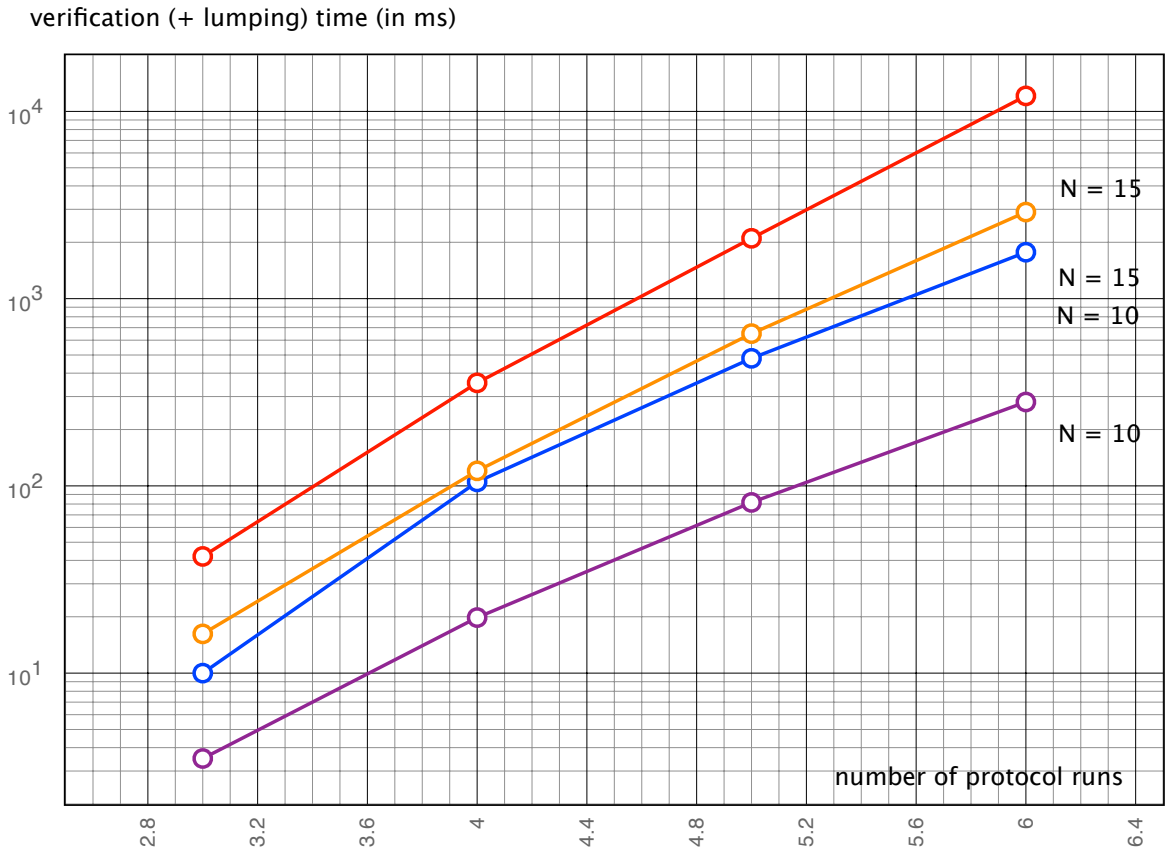
- A protocol for anonymous web browsing (variants: mCrowds, BT-Crowds)
- Hide user's communication by random routing within a crowd
  - sender selects a crowd member randomly using a uniform distribution
  - selected router flips a biased coin:
    - with probability $1 - p$: direct delivery to final destination
    - otherwise: select a next router randomly (uniformly)
  - once a routing path has been established, use it until crowd changes
- Rebuild routing paths on crowd changes (R times)
- Probable innocence:
  - probability real sender is discovered $< \frac{1}{2}$ if $N \geqslant \frac{p}{p-\frac{1}{2}} \cdot (c+1)$
  - where N is crowd's size and c is number of corrupt crowd members

Crowds protocol



state space reductions for eventually observer the real sender more than once

Crowds protocol



run times for eventually observer the real sender more than once

It mostly pays off!

- Significant state space reductions
  - reduction factors varying from 0 to 3 orders of magnitude
  - property-driven bisimulation yields better results
  - . . . even after symmetry reduction
- Mostly a reduction of the total verification time
  - depends on “denseness” and structure of the Markov chain
  - long run: convergence rate of numerical computations
  - reward models: huge reductions of verification time (up to 4 orders)
- Possibility to exploit component-wise minimisation

Overview of the tutorial

- Introduction
- Discrete-time probabilistic models
  - fully probabilistic systems: discrete-time Markov chains
  - probabilistic CTL, verification algorithms, counterexamples
- Continuous-time probabilistic models
  - fully probabilistic systems: continuous-time Markov chains
  - stochastic CTL, verification algorithms, experimental results
  - exhibiting non-determinism: continuous-time MDPs
- Stochastic cost models (not today)
  - discrete-time and continuous time Markov reward models

CTMCs

A labeled CTMC $\mathcal{C}$ is a quadruple $(S, \mathbf{P}, E, L)$ with:

$E : S \to \mathbb{R}_{\geqslant 0}$ is the *exit-rate* function

- $E(s)^{-1}$ is the average residing time in s

- Interpretation: $\mathbf{P}(s, s', t) = \mathbf{P}(s, s') \cdot \left(1 - e^{-E(s) \cdot t}\right)$
  - the probability to move from state s to s' within t time units
  - $1 - e^{-E(s) \cdot t}$ is the probability to take *some* outgoing transition from s within $[0, t]$

- $(S, \mathbf{P}, L)$ is the *embedded* DTMC of $\mathcal{C}$

$\Rightarrow$ a CTMC is a Kripke structure with randomly timed transitions

Exponential distribution

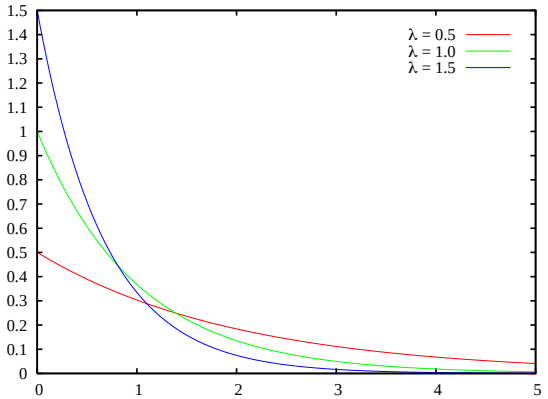
Continuous r. v. X is *exponential* with parameter $\lambda > 0$ if its density is

$$f_X(x) = \lambda \cdot e^{-\lambda \cdot x} \quad \text{for } x > 0 \quad \text{and } 0 \text{ otherwise}$$

Cumulative distribution of X:

$$F_X(d) = \int_0^d \lambda \cdot e^{-\lambda \cdot x}\, dx = 1 - e^{-\lambda \cdot d}$$

- expectation $E[X] = \int_0^\infty x \cdot \lambda \cdot e^{-\lambda \cdot x}\, dx = \frac{1}{\lambda}$
- variance $Var[X] = \frac{1}{\lambda^2}$



density function

Exponential distributions

- Are *adequate* for many real-life phenomena
  - the time until you have your next car accident (failure rates)
  - inter-arrival times of jobs, telephone calls, and so on
- Are *memoryless*: $\Pr\{X > t + d \mid X > t\} = \Pr\{X > d\}$, moreover:
  - $\Pr\{X = \min(X, Y)\} = \frac{\lambda}{\lambda+\mu}$
- Can *approximate* general distributions arbitrarily closely
- *Maximal entropy* probability distribution if only the mean is known

Modelling techniques for CTMCs

- Stochastic Petri nets (Molloy 1977)
- Markovian queueing networks (Kleinrock 1975)
- Stochastic automata networks (Plateau 1985)
- Stochastic activity networks (Meyer & Sanders 1985)
- Stochastic process algebra (Herzog *et al.*, Hillston 1993)
- Probabilistic input/output automata (Smolka *et al.* 1994)

and many variants thereof

Stochastic CTL (Aziz et al., 1996, Baier et al., 1999)

State-formulas $\Phi ::= a \mid \neg\Phi \mid \Phi \vee \Phi \mid \mathbb{S}_J(\Phi) \mid \mathbb{P}_J(\varphi)$

$\mathbb{S}_J(\Phi)$ probability that Φ holds in steady state lies $\in J$

$\mathbb{P}_J(\varphi)$ probability that paths fulfill φ lies $\in J$

Path-formulas $\varphi ::= \Phi \,\mathsf{U}^I\, \Phi$ with interval I

$\Phi \,\mathsf{U}^I\, \Psi$ Φ holds along the path until Ψ holds at time $t \in I$

CTL operator U is a special cases, viz. for $I = [0, \infty)$

Example properties

- In $\geqslant$ 92% of the cases, a goal state is legally reached within 3.1 sec:

$$\mathbb{P}_{\geqslant 0.92}\left(\neg\, \mathit{illegal}\ \mathsf{U}^{\leqslant 3.1}\ \mathit{goal}\right)$$

- . . . a state is soon reached guaranteeing 0.9999 long-run availability:

$$\mathbb{P}_{\geqslant 0.92}\left(\neg\, \mathit{illegal}\ \mathsf{U}^{\leqslant 0.7}\ \mathbb{S}_{\geqslant 0.9999}\left(\mathit{goal}\right)\right)$$

- On the long run, illegal states can (almost surely) not be reached in the next 7.2 time units:

$$\mathbb{S}_{\geqslant 0.9999}\left(\mathbb{P}_{\geqslant 1}\left(\Box^{\leqslant 7.2}\neg\, \mathit{illegal}\right)\right)$$

Semantics of CSL: state-formulas

$\mathcal{C}, s \models \Phi$ if and only if formula Φ holds in state s of CTMC $\mathcal{C}$

Relation $\models$ is defined by:

$$
\begin{array}{lll}
s \models a & \text{iff} & a \in L(s) \\
s \models \neg\, \Phi & \text{iff} & \text{not } (s \models \Phi) \\
s \models \Phi \ \vee \ \Psi & \text{iff} & (s \models \Phi) \text{ or } (s \models \Psi) \\
s \models \mathbb{S}_{J}(\Phi) & \text{iff} & \lim_{t\to\infty} \Pr\{\, \sigma \in \mathit{Paths}(s) \mid \sigma@t \models \Phi \,\} \ \in \ J \\
s \models \mathbb{P}_{J}(\varphi) & \text{iff} & \Pr\{\, \sigma \in \mathit{Paths}(s) \mid \sigma \models \varphi \,\} \ \in \ J
\end{array}
$$

$\Pr\{\ldots\}$ is measurable by a (i.e., cone) Borel space construction on paths in a CTMC

Semantics of CSL: path-formulas

A *path* σ in CTMC $\mathcal{C}$ is an infinite alternating sequence

$$s_0\, t_0\, s_1\, t_1 \ldots \quad \text{with} \quad \mathbf{R}(s_i, s_{i+1}) > 0 \quad \text{and} \quad t_i \geqslant 0$$

non time-divergent paths have probability zero

Semantics of path-formulas is defined by:

$$\sigma \models \Phi \,\mathsf{U}^I\, \Psi \quad \text{iff } \exists t \in I.\ ((\forall t' \in [0,t).\, \sigma@t' \models \Phi) \wedge \sigma@t \models \Psi)$$

where $\sigma@t$ denotes the state in the path σ at time t

Model-checking CSL

- Check which states in a CTMC satisfy a CSL formula:
  - compute recursively the set $Sat(\Phi)$ of states that satisfy Φ

  $\Rightarrow$ recursive descent computation over the parse tree of Φ

- For the non-stochastic part: as for CTL
- For all probabilistic formulae not involving a time bound: as for PCTL
  - using the embedded DTMC
- How to compute $Sat(\Phi)$ for the timed and $\mathbb{S}$ operators?

Model-checking the steady-state operator

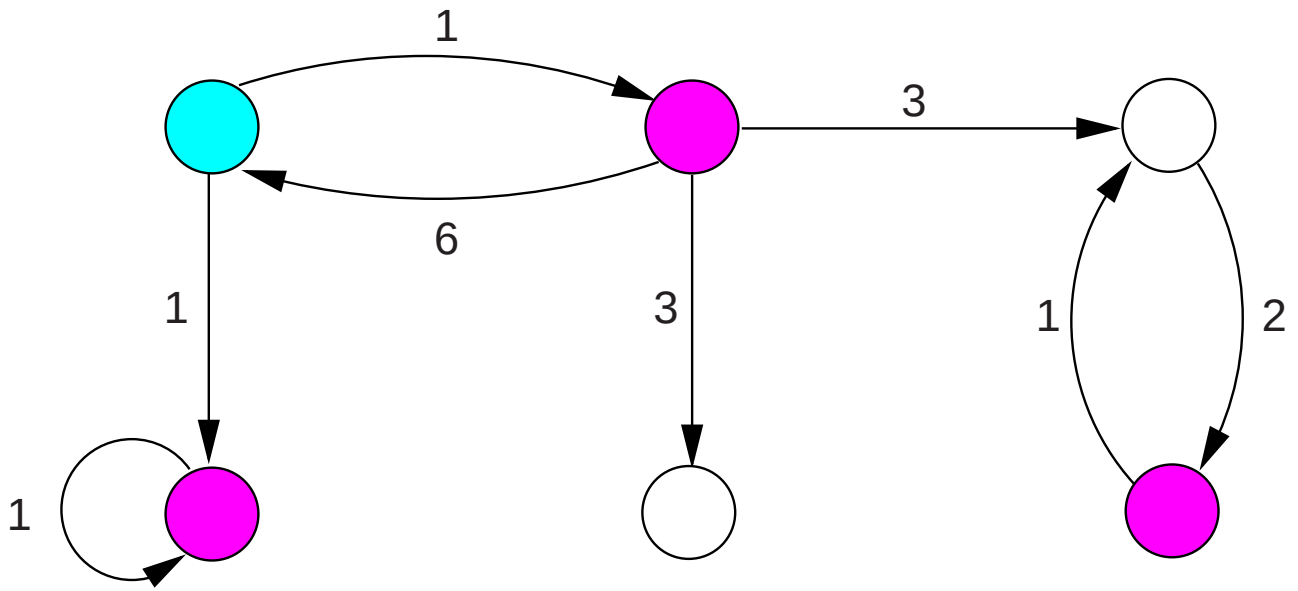
- For an ergodic (i.e., strongly-connected) CTMC:

$$s \in Sat(\mathbb{S}_{\trianglelefteq p}(\Phi)) \text{ iff } \sum_{s' \in Sat(\Phi)} \pi_{s'} \trianglelefteq p$$

  $\Longrightarrow$ this boils down to a standard steady-state analysis

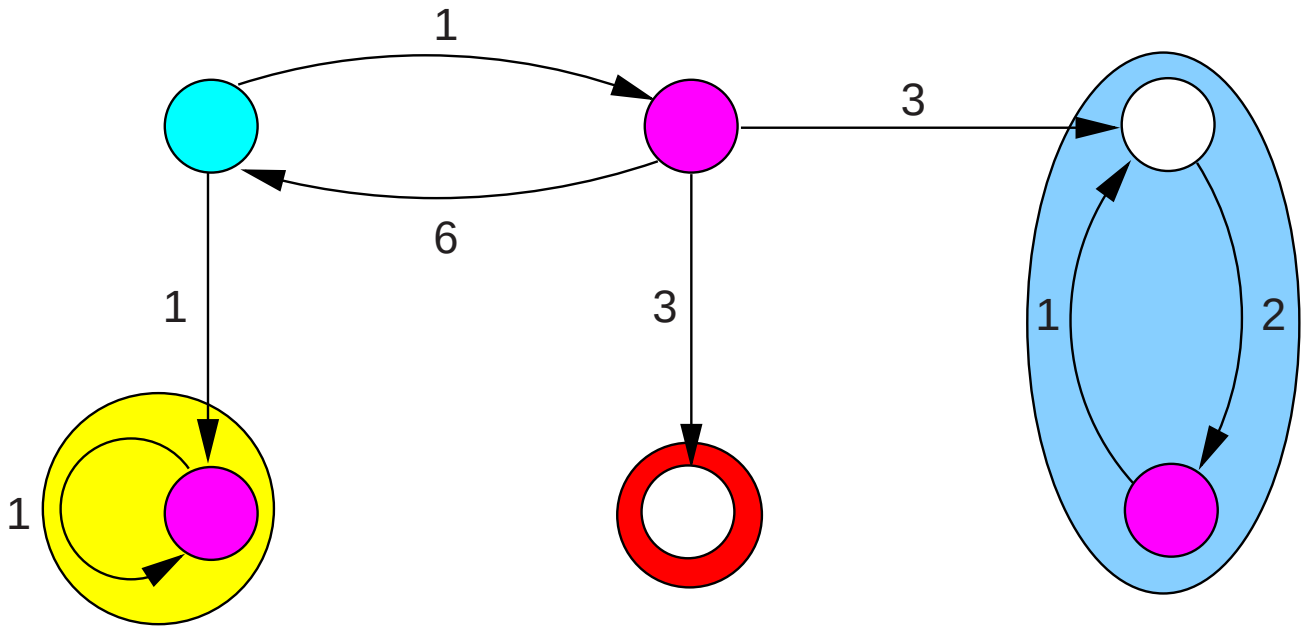
- For an arbitrary CTMC:
  - determine the *bottom* strongly-connected components (BSCCs)
  - for BSCC B determine the steady-state probability of a Φ-state
  - compute the probability to reach BSCC B from state s
  - check whether $\sum_B \left(\Pr\{ \text{ reach } B \text{ from } s \} \cdot \sum_{s' \in B \cap Sat(\Phi)} \pi^B_{s'} \right) \trianglelefteq p$

Verifying steady-state properties: an example



determine the bottom strongly-connected components

Verifying steady-state properties: an example



$s \models \mathbb{S}_{>0.75}(\textit{magenta})$ iff $Prob(s, \Diamond at_{yellow}) \cdot \pi^{yellow}(\textit{magenta}) + Prob(s, \Diamond at_{blue}) \cdot \pi^{blue}(\textit{magenta}) > 0.75$

Checking time-bounded reachability

- $s \models \mathbb{P}_{J}(\Phi \,\mathsf{U}^{\leqslant t}\, \Psi)$ if and only if $Prob(s, \Phi \,\mathsf{U}^{\leqslant t}\, \Psi) \in J$
- $Prob(s, \Phi \,\mathsf{U}^{\leqslant t} \Psi)$ is the least solution of: (Baier, Katoen & Hermanns, 1999)
  - 1 if $s \models \Psi$
  - if $s \models \Phi \ \wedge \ \neg\Psi$:

$$\int_0^t \sum_{s' \in S} \underbrace{\mathbf{P}(s,s') \cdot E(s) \cdot e^{-\mathbf{E}(s)\cdot x}}_{\substack{\text{probability to move to} \\ \text{state } s' \text{ at time } x}} \cdot \underbrace{Prob(s', \Phi \,\mathsf{U}^{\leqslant t-x}\, \Psi)}_{\substack{\text{probability to fulfill } \Phi \,\mathsf{U}\, \Psi \\ \text{before time } t-x \text{ from } s'}} \; dx$$

  - 0 otherwise

Reduction to transient analysis

(Baier, Haverkort, Hermanns & Katoen, 2000)

- Make all Ψ- and all $\neg(\Phi \vee \Psi)$-states absorbing in $\mathcal{C}$
- Check $\Diamond^{=t}\,\Psi$ in the obtained CTMC $\mathcal{C}'$
- This is a standard transient analysis in $\mathcal{C}'$:

$$\sum_{s' \models \Psi} \Pr\{\sigma \in \mathit{Paths}(s) \mid \sigma@t = s'\}$$

  - compute by solving linear differential equations, or discretization

$\Rightarrow$ Discretization + matrix-vector multiplication + Poisson probabilities

Interval-bounded reachability

- For any path σ that fulfills $\Phi \; \mathsf{U}^{[t,t']} \; \Psi$ with $0 < t \leqslant t'$:
  - Φ holds continuously up to time t, and
  - the suffix of σ starting at time t fulfills $\Phi \; \mathsf{U}^{[0,t'-t]} \; \Psi$
- Approach: divide the problem into two:

$$\underbrace{\sum_{s' \models \Phi} \pi^{\mathcal{C}'}(s, s', t)}_{\text{check } \Box^{[0,t]} \, \Phi} \cdot \underbrace{\sum_{s'' \models \Psi} \pi^{\mathcal{C}''}(s', s'', t'-t)}_{\text{check } \Phi \; \mathsf{U}^{[0,t'-t]} \; \Psi \text{ with starting distribution } \underline{\pi}^{\mathcal{C}'}(t)}$$

  - where $\mathcal{C}'$ equals $\mathcal{C}$ with all Φ-states absorbing
  - and $\mathcal{C}''$ equals $\mathcal{C}$ with all Ψ and $\neg\,(\Phi \vee \Psi)$-states absorbing

Summary of model checking CSL

(Aziz *et al.*, 1996), (Baier *et al.*, 1999), (Katoen *et al.*, 2001)

- Recursive descent over the parse tree of Φ
- Steady-state operator: graph analysis + linear system(s) of equations
- Time-bounded until: model transformation and uniformisation
- Worst case time-complexity: $\mathcal{O}(|\Phi|\cdot(|\mathbf{R}|\cdot q\cdot t_{max} + |S|^{2.81}))$

  with $|\Phi|$ the length of Φ, uniformisation rate q, t_{max} the largest time bound in Φ
- Tools:

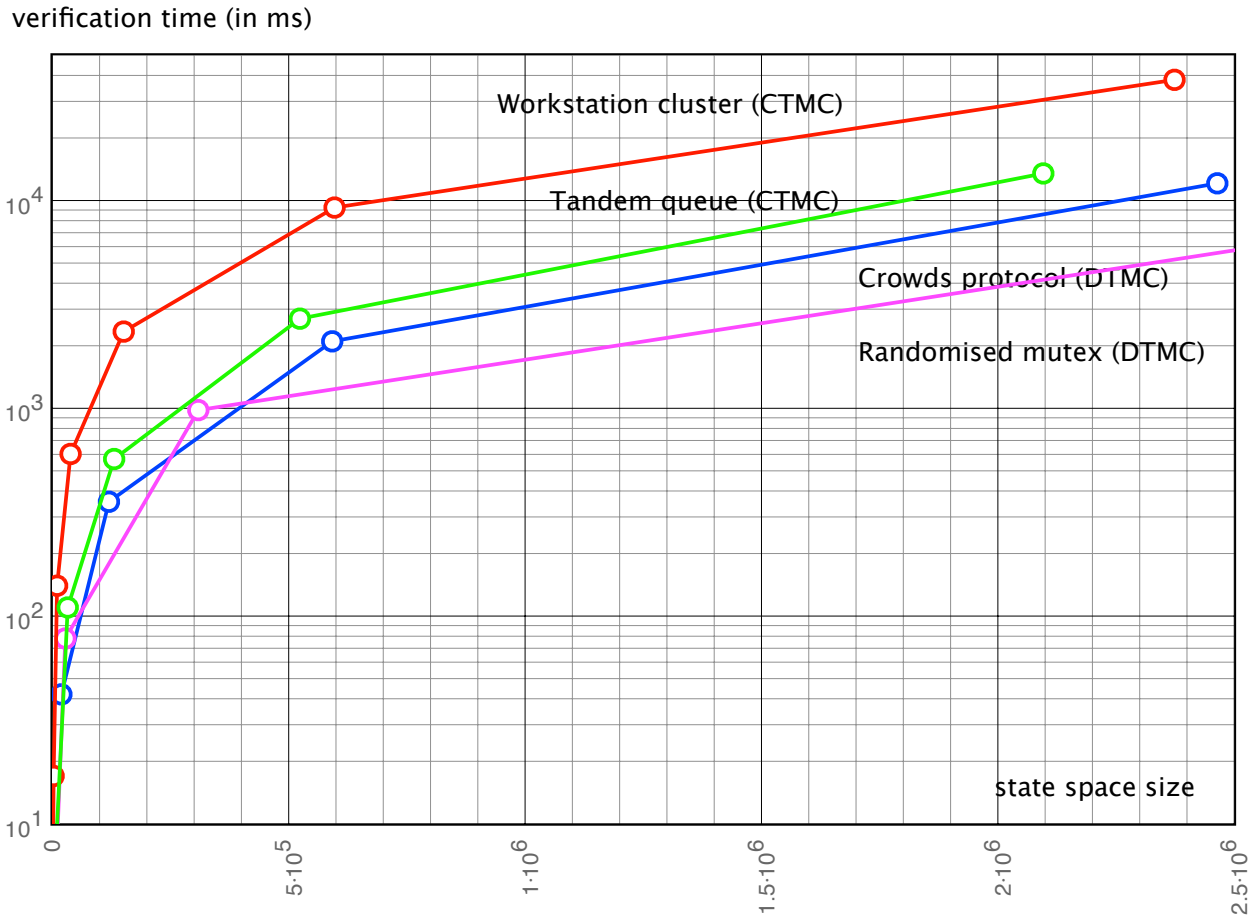
  E $\vdash$ MC^2 , PRISM, APNN TOOLBOX, MRMC, 2TOWERS, YMER*, VESTA*

Markov reward model checker (MRMC)

(Zapreev & Meyer-Kayser, 2000/2005)

- Supports DTMCs, CTMCs and cost-based extensions thereof
  - temporal logics: P(R)CTL and CS(R)L
  - bounded until, long run properties, and interval bounded until
- Sparse-matrix representation
- Command-line tool (in C)
  - experimental platform for new (e.g., reward) techniques
  - back-end of GreatSPN, PEPA WB, PRISM and stochastic GG tool
  - freely downloadable under Gnu GPL license
- Experiments: Pentium 4, 2.66 GHz, 1 GB RAM

Verification times



Overview of the tutorial

- Introduction
- Discrete-time probabilistic models
  - fully probabilistic systems: discrete-time Markov chains
  - probabilistic CTL, verification algorithms, counterexamples
- Continuous-time probabilistic models
  - fully probabilistic systems: continuous-time Markov chains
  - stochastic CTL, verification algorithms, experimental results
  - exhibiting non-determinism: continuous-time MDPs
- Stochastic cost models (not today)
  - discrete-time and continuous time Markov reward models

Scheduling a batch of stochastic jobs

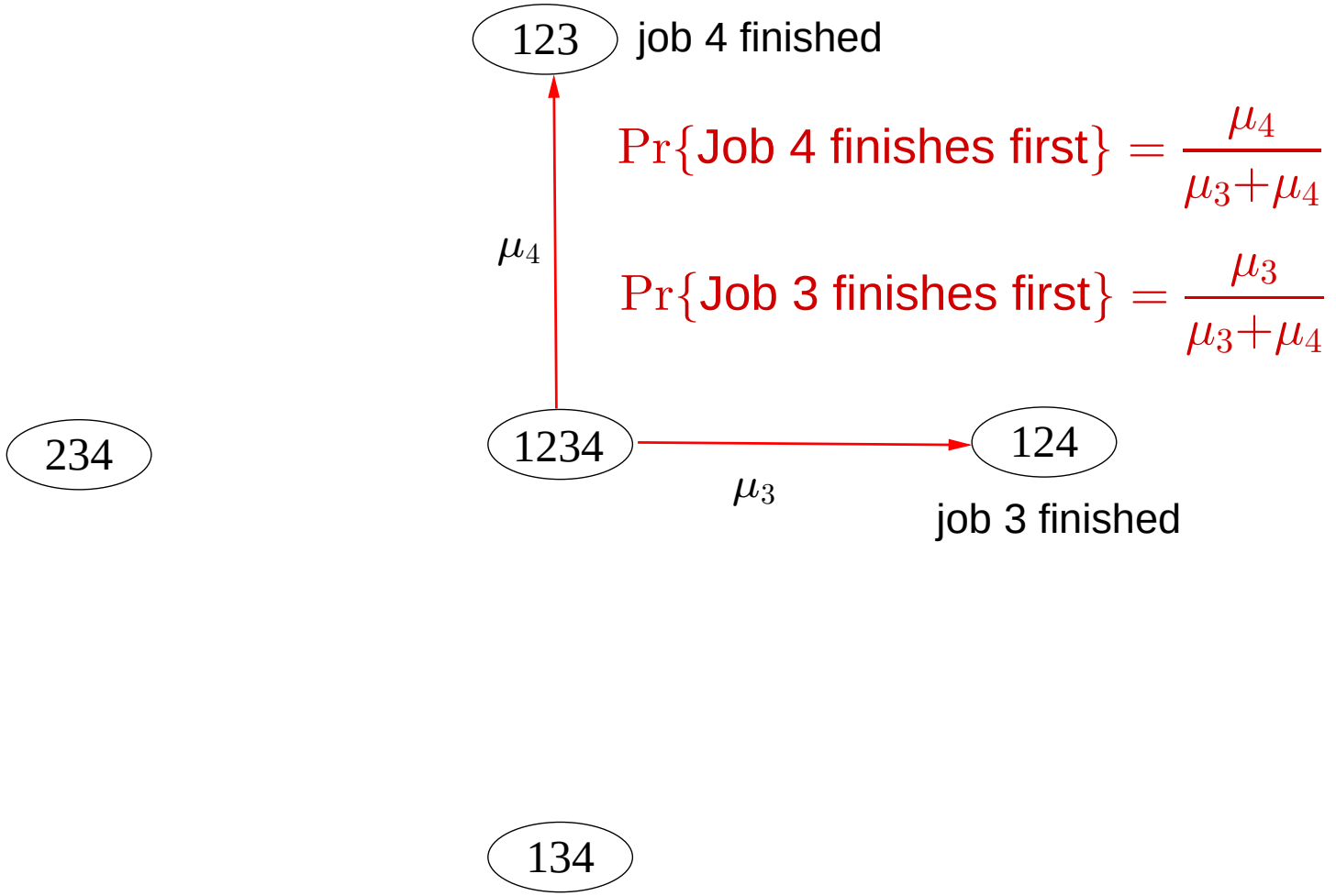
- N jobs with a random duration and M identical machines
  - the job's probability distribution is unknown
  - but its mean duration is known

  $\Rightarrow$ assuming exponential distributions (rate μ_i) is most appropriate

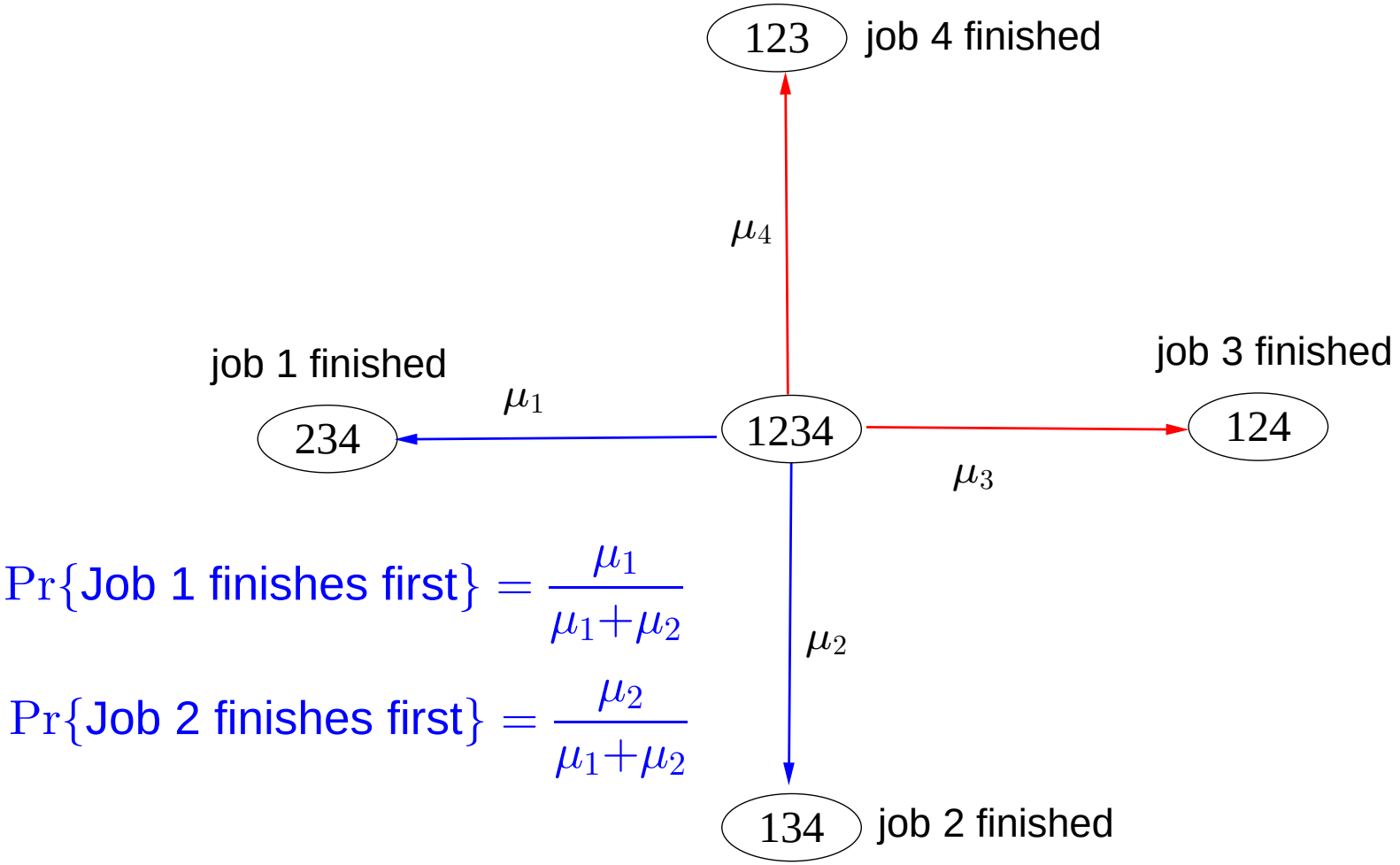
- Jobs maybe preempted at decision epochs

- Which scheduling policy to take? (Bruno *et. al*, 1981)
  - to minimize the expected total flowtime: SEPT policy
  - to minimize the expected finishing time of last job (makespan): LEPT policy
  - to maximize the probability to finish all jobs at time t: unknown

Scheduling a batch of stochastic jobs ($N = 4; M = 2$)



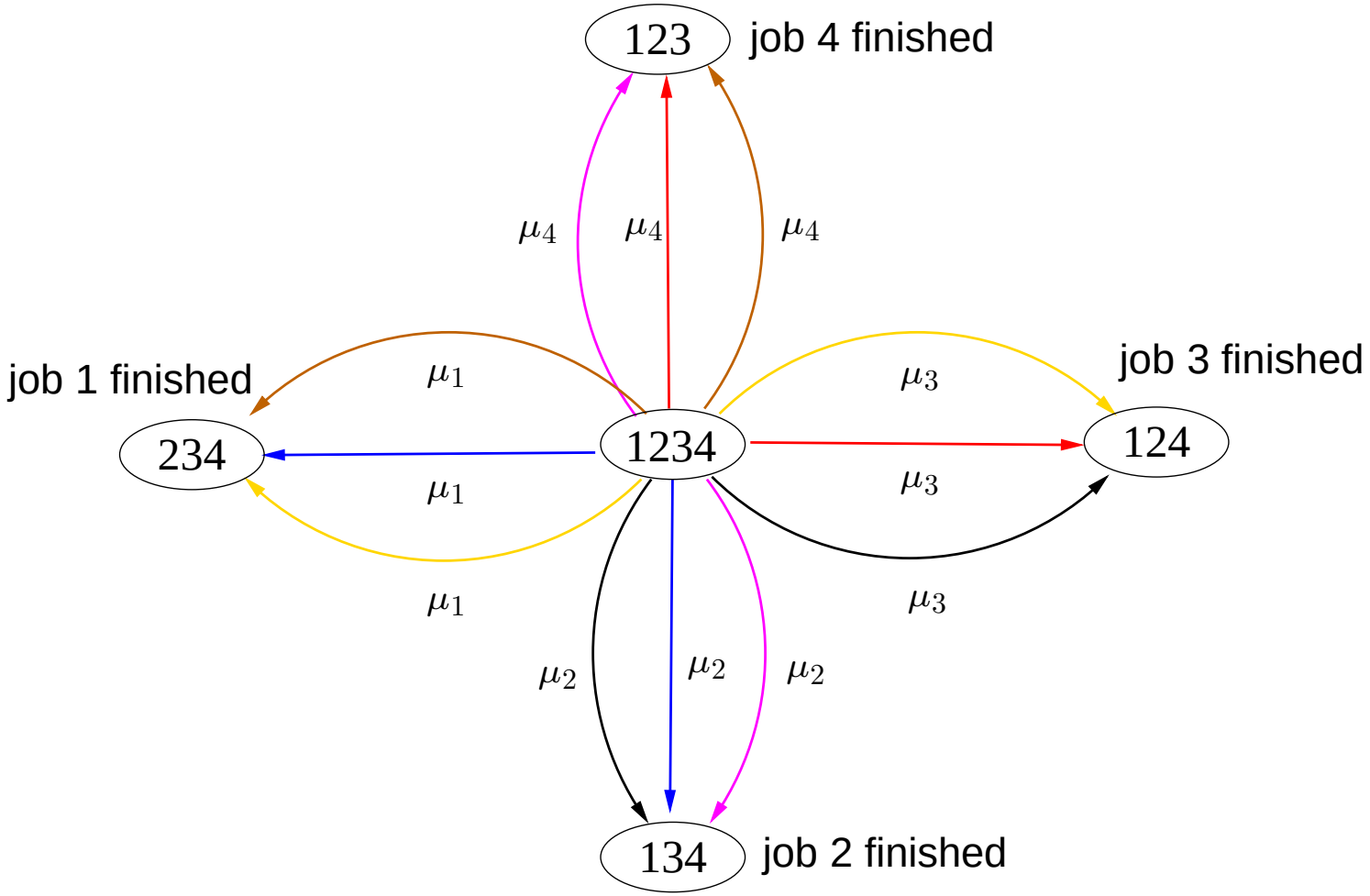
Scheduling a batch of stochastic jobs ($N = 4; M = 2$)



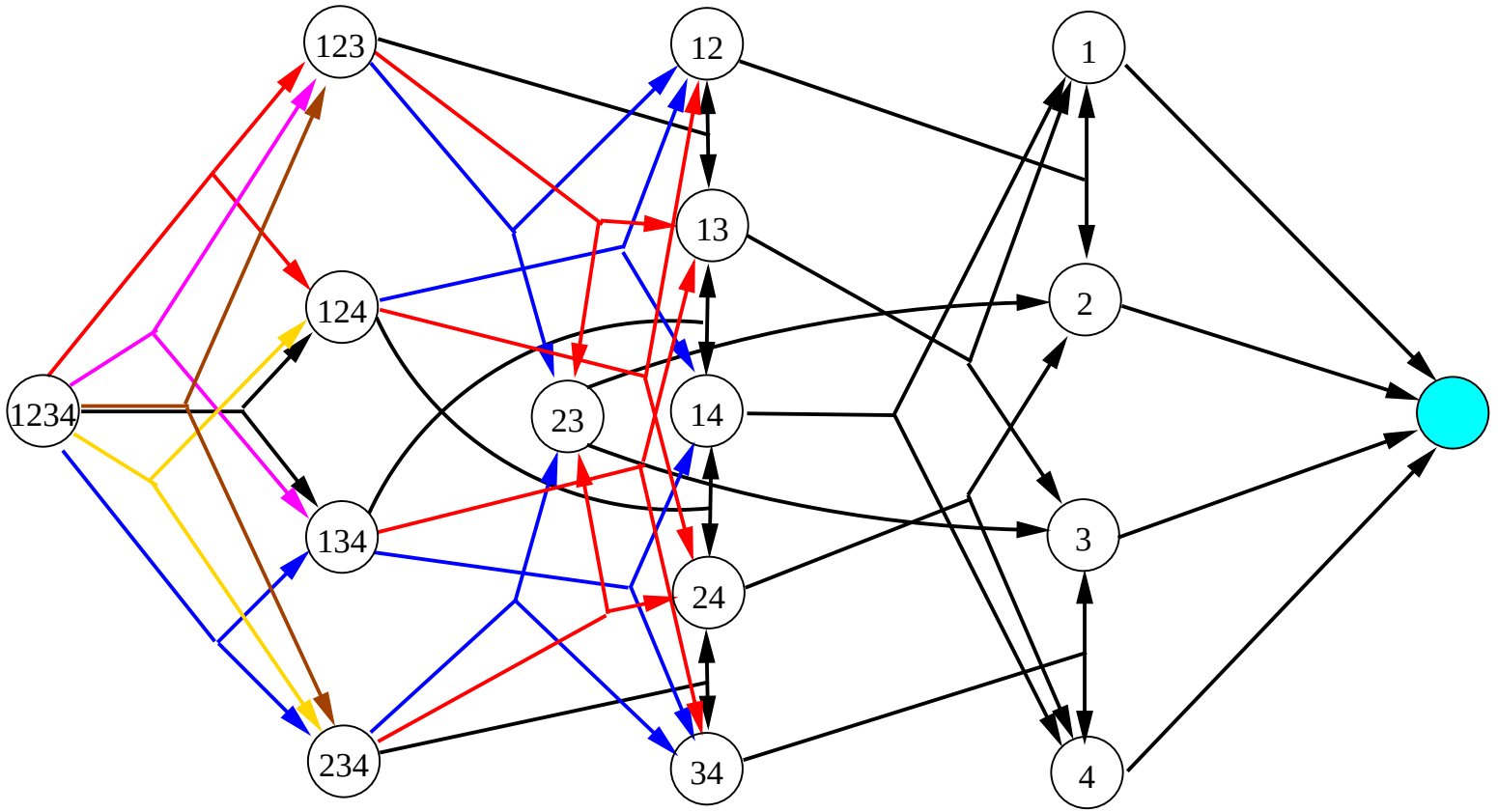
$$\Pr\{\text{Job 1 finishes first}\} = \frac{\mu_1}{\mu_1+\mu_2}$$

$$\Pr\{\text{Job 2 finishes first}\} = \frac{\mu_2}{\mu_1+\mu_2}$$

Scheduling a batch of stochastic jobs ($N = 4; M = 2$)



Scheduling a batch of stochastic jobs ($N = 4; M = 2$)



$\Rightarrow$ this is a continuous-time Markov decision process

Even more reasons for CTMDPs

- They are *a key model* in:
  - stochastic control theory (e.g., dynamic power management)
  - stochastic scheduling
- They are the *semantic basis* for
  - (non-well-specified) stochastic activity networks
  - generalised stochastic Petri nets (with confusion)
  - some Markovian process algebra and stochastic UML statecharts
- Our interests:
  - model-checking Continuous Stochastic Logic (CSL)
  - abstraction-refinement for continuous-time Markov chains
  - stochastic scheduling

Continuous-time MDP

A CTMDP $\mathcal{C}$ is a triple $(S, \mathit{Act}, \mathbf{R})$ with:

- Finite (discrete) set of states S and finite set of actions Act
- A three-dimensional rate matrix $\mathbf{R} : (S \times \mathit{Act} \times S) \to \mathbb{R}_{\geqslant 0}$ such that
  - $\mathbf{R}(s, \alpha, s') = \lambda$ is the rate of a negative exponential distribution

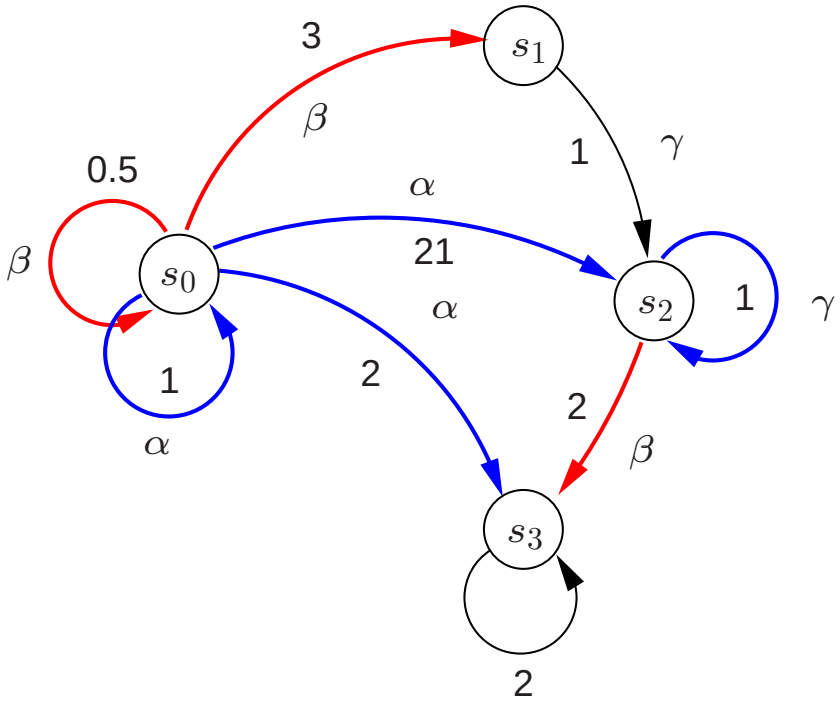
$$F_X(t) = 1 - e^{-\lambda \cdot t}, \quad t \geqslant 0; \qquad E[X] = \frac{1}{\lambda}$$

  such that $\mathit{Act}(s) = \{\, \alpha \mid E(s, \alpha) > 0 \,\} \neq \varnothing$ for all s

  - $E(s, \alpha) = \mathbf{R}(s, \alpha, S) = \sum_{s' \in S} \mathbf{R}(s, \alpha, s')$ be the exit rate to "leave" s via α

$\Rightarrow$ if $|\mathit{Act}| = 1$ there is no non-determinism and we obtain a CTMC

Example

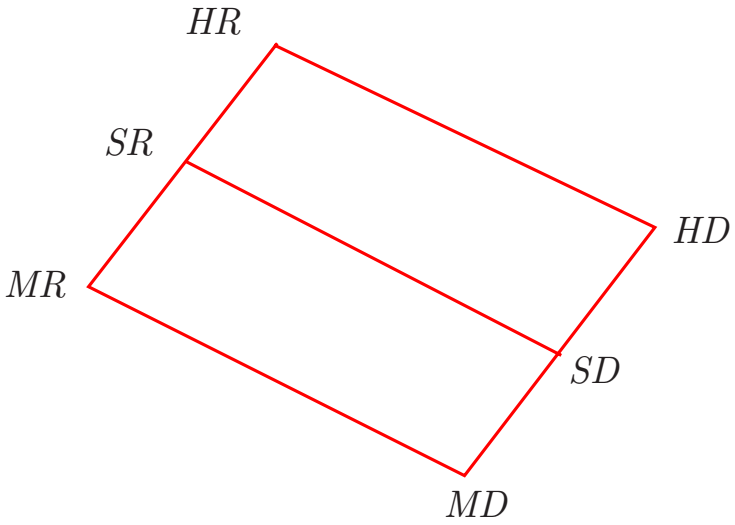


- $\mathbf{R}(s_0, \alpha, s_2) = 21$ and
  $\mathbf{R}(s_0, \alpha, s_0) = 1$
- $E(s_0, \alpha) = 24$ and
  $E(s_0, \beta) = 3.5$
- $\mathbf{P}(s_0, \alpha, s_2) = \frac{7}{8}$ and
  $\mathbf{P}(s_0, \alpha, s_0) = \frac{1}{24}$
- example run:

$$s_0 \xrightarrow{\alpha,2.71} s_0 \xrightarrow{\beta,0.41} s_0 \xrightarrow{\beta,2.1} s_1 \xrightarrow{\gamma,17.2} s_2 \xrightarrow{\gamma,0.2} s_2 \xrightarrow{\beta,\pi} s_3 \ldots$$

Types of schedulers

- Nondeterminism in a CTMDP is resolved by a *scheduler*
  - Available information? — current state (M), # visits (S) or history (H)
  - How to decide? — deterministic (D) or randomized (R)
  - Fairness? — (not today)
- The hierarchy of scheduler classes MD, MR, SD, SR, HD and HR:



alternative terminology: tactic, policy, adversary, . . .

A CTMDP under a HD-scheduler

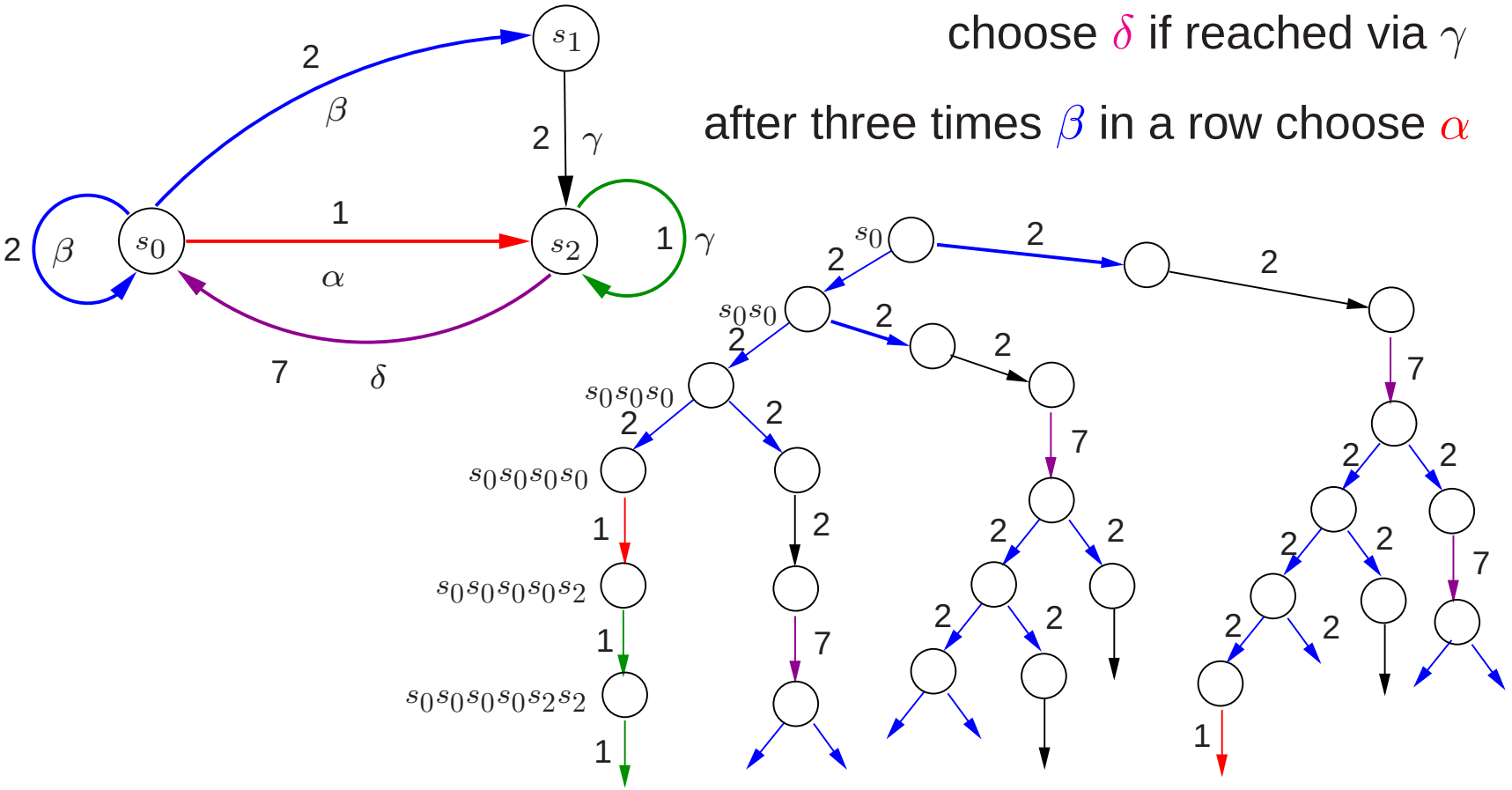
HD-scheduler $D : S^+ \to \mathit{Act}$ on CTMDP $\mathcal{C} = (S, \mathit{Act}, \mathbf{R})$

- Unfold $\mathcal{C}$, resolving the nondeterminism according to D
- Formally, this induces the CTMC $\mathcal{C}_D = (S_D, \mathbf{R}_D)$ with:
  - $S_D = S^+$, unlabeled path fragments $s_0 \to s_1 \to \ldots \to s_{n-1} \to s_n$ in $\mathcal{D}$
  - $\mathbf{R}_D(\sigma, \sigma \to s) = \mathbf{R}(\mathit{last}(\sigma), D(\sigma), s)$ and 0 otherwise
- Mostly infinite, but for MD-schedulers the unfolding is finite:

  all sequences ending in s are lumping equivalent
- The embedded DTMC of $\mathcal{C}_D$ is a tuple $(S_D, \mathbf{P}_D)$ where

  $\mathbf{P}_D(\sigma, \sigma') = \frac{\mathbf{R}_D(\sigma,\sigma')}{E_D(\sigma)}$ if $E_D(\sigma) > 0$ and 0 otherwise

A CTMDP under a HD-scheduler



Problem statement

- CTMDP $(S, \mathit{Act}, \mathbf{R})$ is *uniform* if

$$E(s,\alpha) = E \text{ for some } E > 0 \text{ for any state } s \text{ and any } \alpha \in \mathit{Act}(s)$$

$\Rightarrow$ in a uniformized CTMDP the exit rates for all states and all actions are equal

- For a uniform CTMDP, time $t > 0$, $B \subseteq S$, and $s \in S$, calculate:

$$\sup_{D \in X} \Pr_D(s, \overset{\leqslant t}{\rightsquigarrow} B) \qquad \text{up to some accuracy } \varepsilon$$

  - where $\Pr_D$ denotes the induced probability measure on paths in CTMC $\mathcal{C}_D$

$\Rightarrow$ compute the maximal probability to reach – under a given class of schedulers – a set B of states within t time units when starting in s

Our approach in a nutshell

- Consider approximations of the required probability
- Consider truncated step-dependent (SD) schedulers
  - step-dependent schedulers with a bounded memory (only store first k visits)
- Give a simple and efficient greedy algorithm for such schedulers
- Prove that the computed probabilities coincide with the maximal probabilities under all HD, HR, SD and SR schedulers

Shifting to the embedded DTMC

- For HD-scheduler D, the probabilities in CTMC $\mathcal{C}_D$ are given by:

$$\left(\Pr_D(\sigma, \overset{\leqslant t}{\rightsquigarrow} B)\right)_{\sigma \in S^+} = \sum_{n=0}^{\infty} e^{-E \cdot t} \cdot \frac{(E \cdot t)^n}{n!} \cdot \mathbf{P}_{D,B}^n \cdot \mathbf{i}_B$$

  - $\psi(n) = e^{-E \cdot t} \cdot \frac{(E \cdot t)^n}{n!}$ is the probability of n events occurring within t time units in a Poisson process with rate E
  - where $\mathbf{i}_B(\sigma) = 1$ if and only if *last*$(\sigma) \in B$ and
  - $$\mathbf{P}_{D,B}(\sigma, \sigma') = \begin{cases} \mathbf{P}_D(\sigma, \sigma') & \text{if } \textit{last}(\sigma) \notin B \\ 1 & \text{if } \sigma' = \sigma \text{ and } \textit{last}(\sigma) \in B \\ 0 & \text{otherwise.} \end{cases}$$

A greedy algorithm – basic idea

- Consider truncated SD-schedulers: $D : S \times \{1 \ldots k\} \to \mathit{Act}$
- Construct the optimal one by a "backwards" greedy strategy:
  - for each s choose the action maximizing the probability to reach B in one step:

    select $\mathit{act}(s,k) \in \mathit{Act}(s)$ with $\mathbf{P}(s, \mathit{act}(s,k), B) = \max\limits_{\alpha \in \mathit{Act}(s)} \sum\limits_{s' \in B} \mathbf{P}(s,\alpha,s')$

  - use these actions to calculate the last (k-th) summand of

    $$\left(\sum_{n=0}^{k} \psi(n) \cdot \mathbf{P}_{D,B}^{n} \cdot \mathbf{i}_B \right)(s)$$

  - continue this way for $i = k-1, \ldots, 1$, i.e., choose $\mathit{act}(s,i)$ maximizing the probability to move to a B-state within at most $k-i+1$ steps (under D)

The main recurrence equation

$$
\begin{aligned}
q_i &= \sum_{n=i}^{k} \psi(n) \cdot \mathbf{P}_i \cdot \mathbf{P}_{i+1} \cdot \ldots \cdot \mathbf{P}_n \cdot \mathbf{i}_B \\
&= \psi(i) \cdot \mathbf{P}_i \cdot \mathbf{i}_B \; + \; \sum_{n=i+1}^{k} \psi(n) \cdot \mathbf{P}_i \cdot \mathbf{P}_{i+1} \cdot \ldots \cdot \mathbf{P}_n \cdot \mathbf{i}_B \\
&= \psi(i) \cdot \mathbf{P}_i \cdot \mathbf{i}_B \; + \; \mathbf{P}_i \cdot \sum_{n=i+1}^{k} \psi(n) \cdot \mathbf{P}_{i+1} \cdot \ldots \cdot \mathbf{P}_n \cdot \mathbf{i}_B \\
&= \boxed{\psi(i) \cdot \mathbf{P}_i \cdot \mathbf{i}_B \; + \; \mathbf{P}_i \cdot q_{i+1}}
\end{aligned}
$$

where $q_{k+1} = \mathbf{0}$, the 0-vector

Results

- For any state $s \in S$:

$$\sup_{D \in HD} \Pr_D(s, \overset{\leqslant t}{\rightsquigarrow} B) - \varepsilon \leqslant q(s) \leqslant \sup_{D \in HD} \Pr_D(s, \overset{\leqslant t}{\rightsquigarrow} B)$$

- The supremum under SD-, HR-, and HD-schedulers agree:

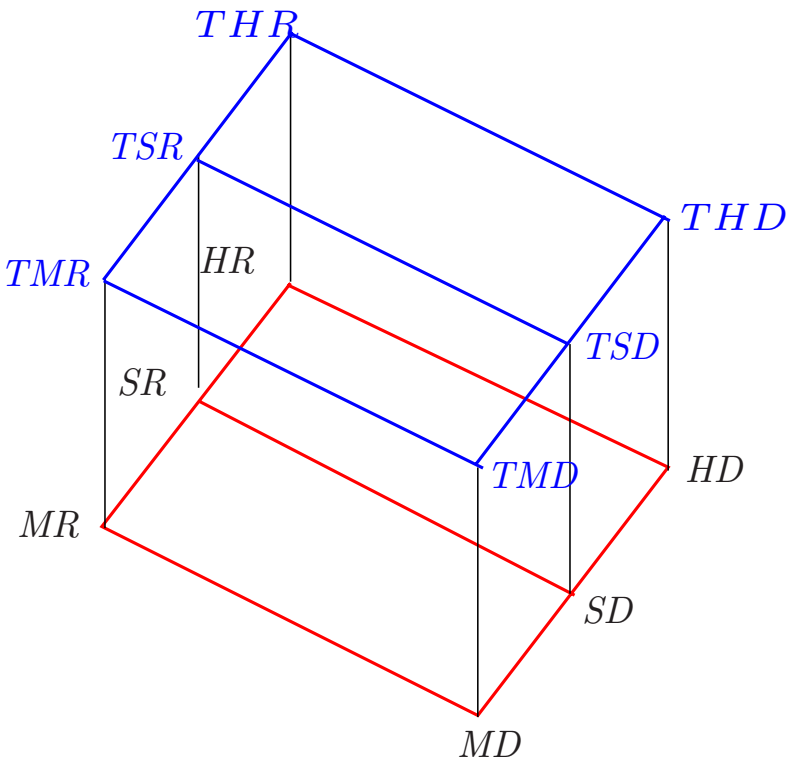
$$\sup_{D \in SD} \Pr_D(s, \overset{\leqslant t}{\rightsquigarrow} B) = \sup_{D \in HD} \Pr_D(s, \overset{\leqslant t}{\rightsquigarrow} B) = \sup_{D \in HR} \Pr_D(s, \overset{\leqslant t}{\rightsquigarrow} B)$$

- Also for SR-schedulers this coincides as $SD \subseteq SR \subseteq HR$
  - for probabilities of other events such correspondence *may not hold* (Beutler & Ross, 1987)

$\Rightarrow$ in general, randomized schedulers may be better than deterministic schedulers

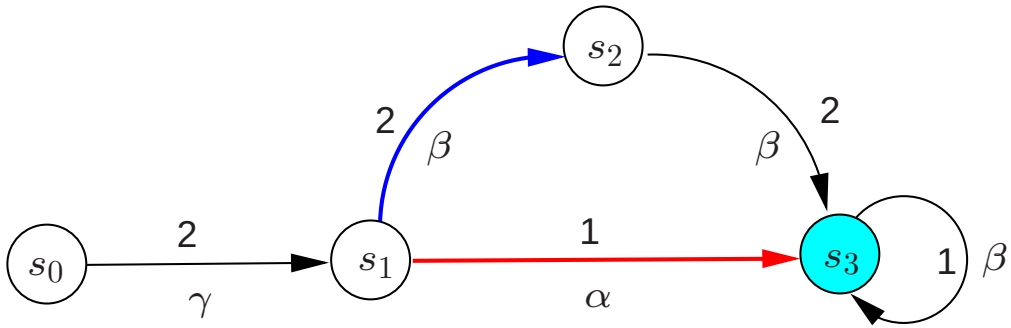
Timed schedulers

- Timed schedulers may make decisions based on the timing of transitions
- A timed-history dependent, deterministic (THD) scheduler:
  - $D : (S \times \mathit{Act} \times \mathbb{R}_{>0})^* \times S \to \mathit{Act}$ with
  - $D(\underbrace{s_0 \overset{\alpha_0,t_0}{\to} \ldots \overset{\alpha_{n-1},t_{n-1}}{\to}}_{\text{timed-history}}, s_n) \in \mathit{Act}(s_n)$



Hierarchy of scheduler classes

A simple example



HD-scheduler D_α chooses α; D_β chooses β

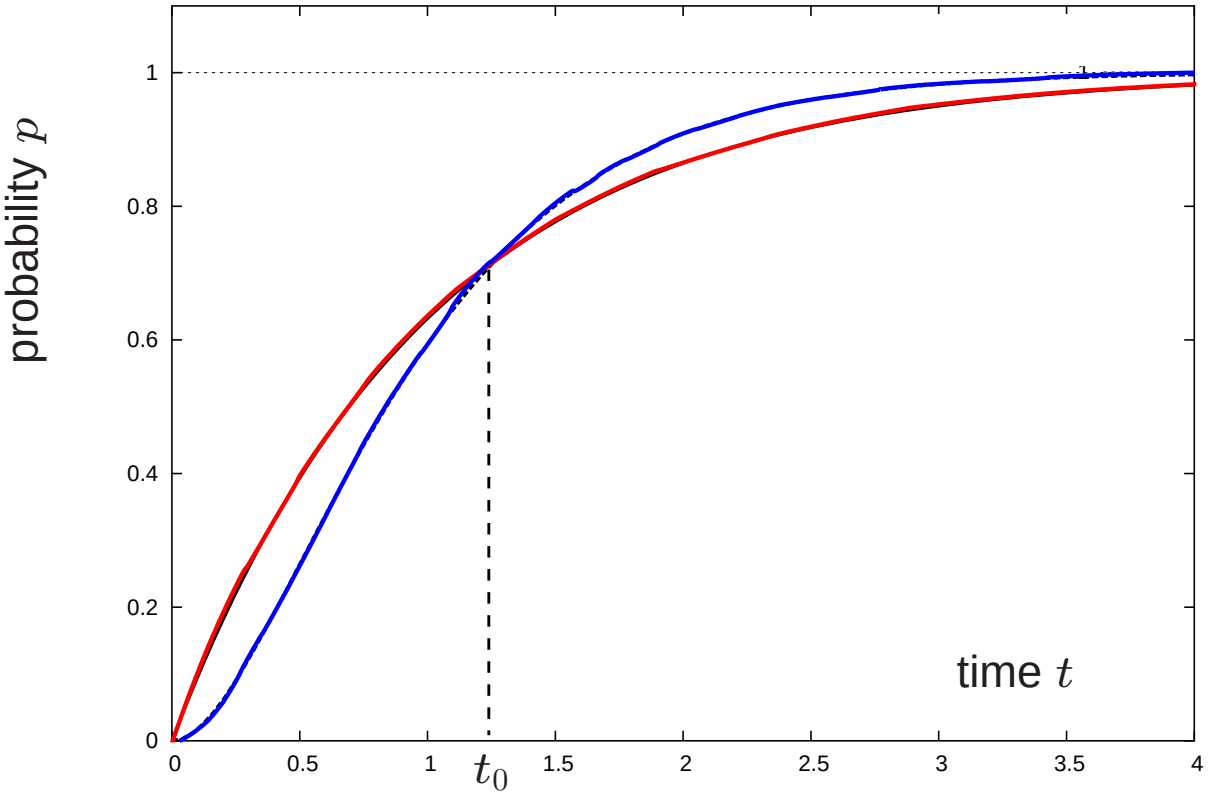
$$\text{For } * \in \{\, \alpha, \beta \,\}: \quad \Pr_{D_*}(s_0, \overset{\leqslant t}{\rightsquigarrow} B) \;=\; \int_0^t 2 \cdot e^{-2(t-x)} \cdot \Pr_{D_*}(s_1, \overset{\leqslant x}{\rightsquigarrow} B)\, dx$$

with

$$\Pr_{D_\alpha}(s_1, \overset{\leqslant t}{\rightsquigarrow} \{\, s_3 \,\}) \;=\; 1 - e^{-t}, \text{ and}$$

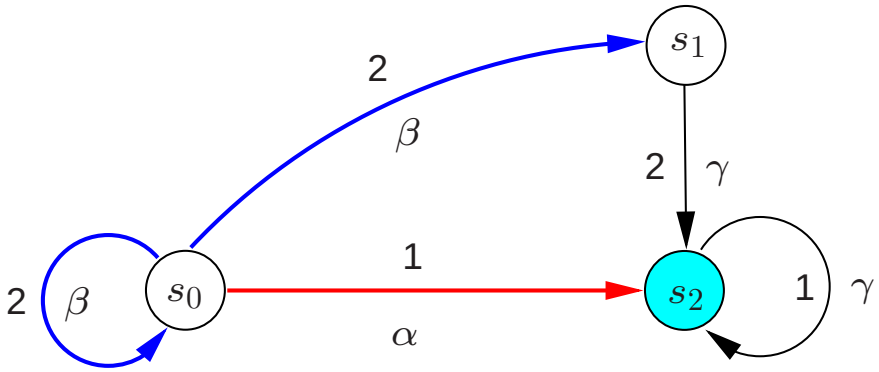
$$\Pr_{D_\beta}(s_1, \overset{\leqslant t}{\rightsquigarrow} \{\, s_3 \,\}) \;=\; 1 - e^{-2t} \cdot (1 + 2t)$$

Timed schedulers may be better



D_α and D_β are outperformed by $D(s_0 \xrightarrow{\gamma,x} s_1) = \begin{cases} \beta & \text{if } t-x \geqslant t_0 \\ \alpha & \text{otherwise} \end{cases}$

Another simple example

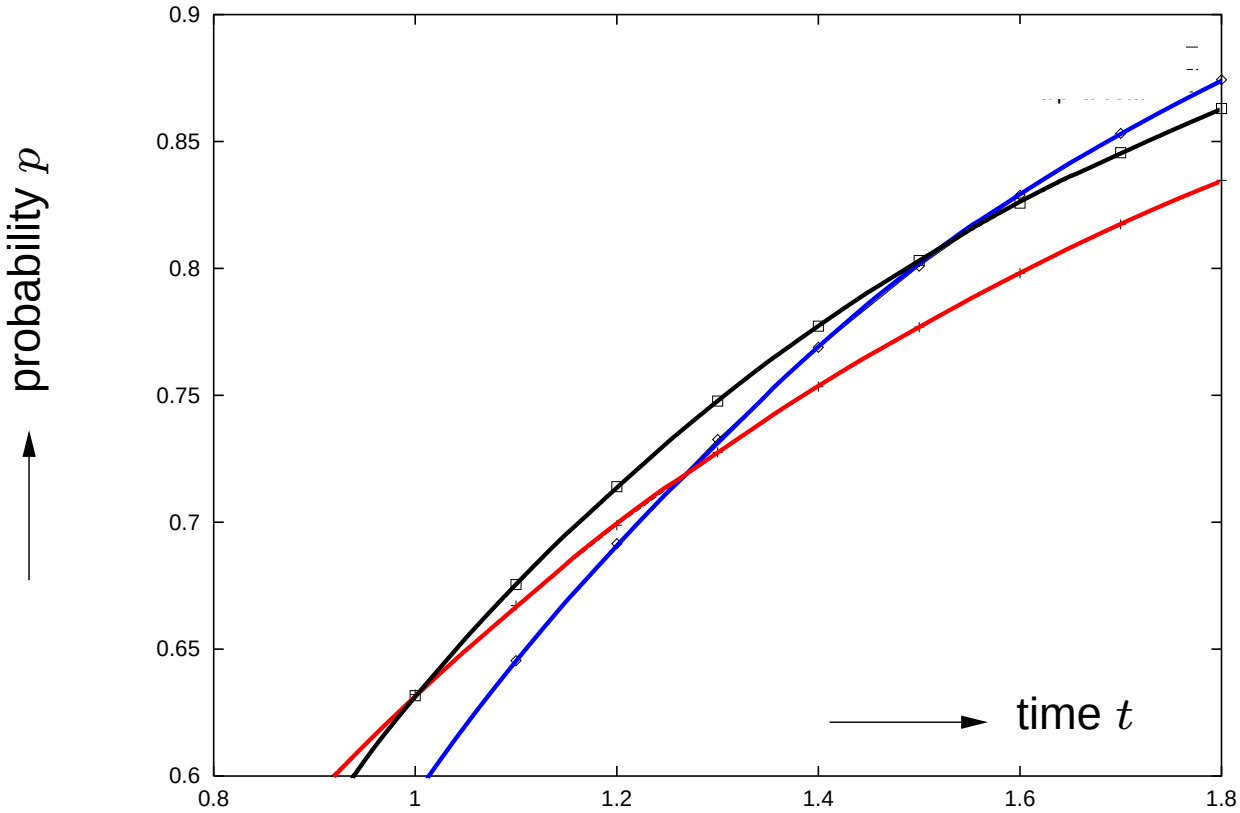


$$\Pr_{D_\alpha}(s_0, \overset{\leqslant t}{\rightsquigarrow} \{ s_2 \}) = 1 - e^{-t}, \text{ and}$$

$$\Pr_{D_\beta}(s_0, \overset{\leqslant t}{\rightsquigarrow} \{ s_2 \}) = 1 - e^{-2t} \cdot (1 + 2t)$$

$\Rightarrow$ If you know the # loops in s_0 the probability may be higher

Simple schedulers may be worse



D_α and D_β are outperformed by $D(s_0, i) = \begin{cases} \beta & \text{if } i \leqslant \lfloor \frac{t-t_0}{4} \rfloor \\ \alpha & \text{otherwise} \end{cases}$

Time-bounded reachability in CTMDPs

- Problem:
  - compute the maximal (or minimal) probability to reach a set of states within t time units in a (uniform) CTMDP
- Results:
  - simple greedy algorithm for truncated step-dependent schedulers
  - SD, SR, (fair) HD, and HR schedulers perform exactly the same
  - MD schedulers are “worse” and timed schedulers may be more optimal
- Space and time-complexity:
  - space complexity in $\mathcal{O}\left(N^2 \cdot M + N\right)$, where $N = \#$ states, $M = \#$ actions
  - worst-case time complexity in $\mathcal{O}(E \cdot t \cdot N^2 \cdot M)$ where E is maximal exit rate
  - worst-case time complexity in $\mathcal{O}(E \cdot t \cdot N^2)$ for timed reachability in CTMCs

  $\Rightarrow$ price to pay is the amount of nondeterminism, **but no more!**

Probabilistic model checking

- . . . . . is a mature automated technique
- . . . . . broad range of applications
- . . . . . is supported by powerful software tools
- . . . . . significant efficiency gain
- . . . . . promising abstraction techniques
- . . . . . possibilitity of counterexamples