Probabilistic Model Checking (2)

GLOBAN Summerschool

Joost-Pieter Katoen

Software Modeling and Verification Group

affiliated to University of Twente, Formal Methods and Tools

Warsaw University, September 25, 2008

Probabilistic models

	Nondeterminism no	Nondeterminism yes
Discrete time	discrete-time Markov chain (DTMC)	Markov decision process (MDP)
Continuous time	CTMC	CTMDP

Reachability probabilities

	Nondeterminism no	Nondeterminism yes
Reachability	linear equation system DTMC	linear programming MDP
Timed reachability	transient analysis (+ uniformization) CTMC	greedy backward reachability uniform CTMDP

Content of this lecture

- Markov decision processes
  - motivation, definition, policies
- Reachability probabilities
  - quantitative and qualitative reachability
- Probabilistic CTL
  - syntax, semantics, model checking

Content of this lecture

⇒ Markov decision processes

- motivation, definition, policies

• Reachability probabilities

- quantitative and qualitative reachability

• Probabilistic CTL

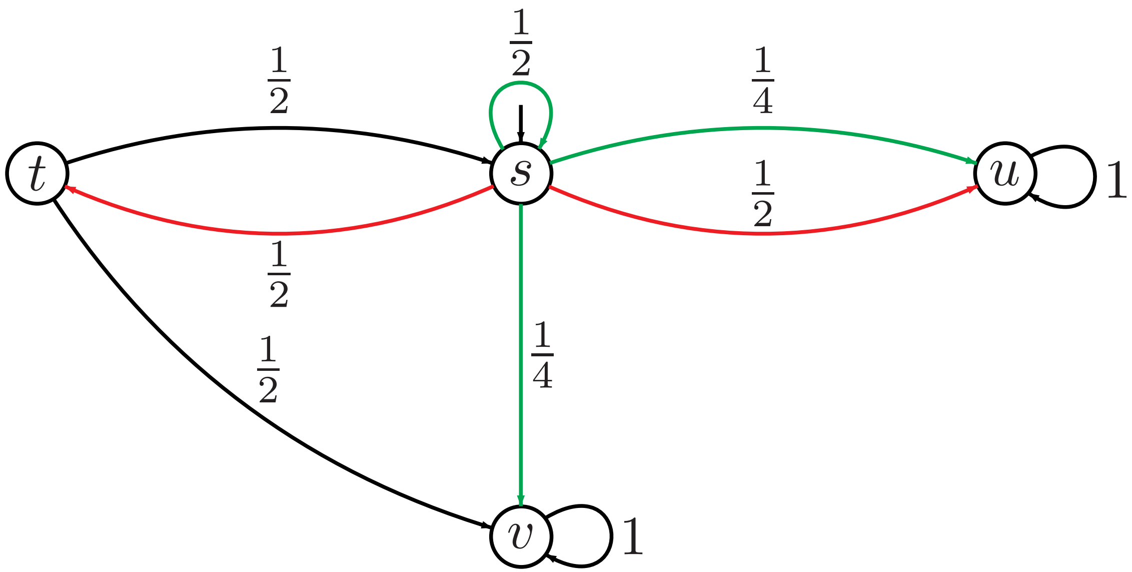
- syntax, semantics, model checking

The importance of nondeterminism

- Implementation freedom as a specification
  - describes *what* the system should do, not *how* it must be implemented
  - leaves freedom for implementation $\Rightarrow$ represent choice by nondeterminism
- Scheduling freedom
  - no info about relative speeds of components yields interleaving model
  - scheduling freedom = which component should move next?
- External environment
  - do not stipulate how the environment will behave
- Incomplete information

Tony Hoary: "There is nothing mysterious about nondeterminism, it arises from the deliberate decision to ignore the factors which influence the selection"

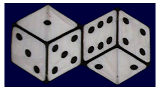
Markov Decision Process



Markov Decision Process

A (labeled) MDP $\mathcal{M} = (S, \mathit{Act}, \mathbf{P}, \iota_{init}, \mathit{AP}, L)$ where

- S is a finite set of states
- *Act* is a finite set of actions
- $\mathbf{P} : S \times \mathit{Act} \times \mathit{Distr}(S)$, transition probability function
- $\iota_{init} \in \mathit{Distr}(S)$, initial state distribution
- $L : S \to 2^{\mathit{AP}}$, state labeling



Motivation

- Stochastic control theory
- Planning and Artificial Intelligence
  - controlled queuing systems, logistics
- Concurrency theory
  - asynchronous communication, channel systems
- Distributed algorithms
  - “local” randomness with concurrent processes

Asynchronous leader election

- An unidirectional asynchronous ring of $N > 2$ nodes
  - each process behaves asynchronously

  $\Rightarrow$ this interleaved concurrency gives rise to an MDP!

- Each node is initially active and proceeds as follows:
  - flip a fair coin (0 and 1), and pass the outcome to your right neighbour
  - if you have chosen 0 while your left neighbour has passed 1, become inactive
  - send a counter around the ring: if only active node $\Rightarrow$ become leader

(Itai & Rodeh, 1990)

Pseudo-code for a single process

$$
\begin{array}{l}
\textit{mode}_i := \textit{active};\\
\textbf{do} \quad :: \textit{mode}_i = \textit{active} \Rightarrow\\
\qquad\qquad x_i := \texttt{random}(0,1);\\
\qquad\qquad c_{i+1}!x_i; c_i?y_i;\\
\qquad\qquad \textbf{if} \quad :: \; y_i = 1 \wedge x_i = 0 \quad \Rightarrow \quad \textit{mode}_i := \textit{passive};\\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \#\textit{active} := \#\textit{active} - 1\\
\qquad\qquad\qquad :: \; y_i = 0 \vee x_i = 1 \quad \Rightarrow \quad \texttt{skip}\\
\qquad\qquad \textbf{fi}\\
\qquad :: \textit{mode}_i = \textit{passive} \Rightarrow \; c_i?y_i; \, c_{i+1}!y_i\\
\textbf{od}
\end{array}
$$

Policies

- Decisions of a policy are either deterministic (D) or randomized (R)

- $\mathfrak{S} : S^+ \to$ *Act* is a history-deterministic (HD) policy with

$$\mathfrak{S}(\underbrace{s_0\, s_1 \ldots s_n}_{\text{history}}) \;\in\; \underbrace{\{\, \alpha \mid \exists s \in S.\, \mathbf{P}(s_n, \alpha, s) > 0 \,\}}_{Act(s_n)}$$

note: actions are not part of the history since $\alpha_{i+1} = \mathfrak{S}(s_0 \ldots s_i)$

- $\mathfrak{S}$ is memoryless (M) if in a state always the same decision is taken

every M-policy is an H-policy; not the converse

alternative terminology: adversary, scheduler, tactic, strategy, . . .

Policies

- Decisions of a policy are either deterministic (D) or randomized (R)
- $\mathfrak{S} : (S \times \mathit{Act})^* \times S \rightarrow \mathit{Distr}(\mathit{Act})$ is a history-randomized (HR) policy

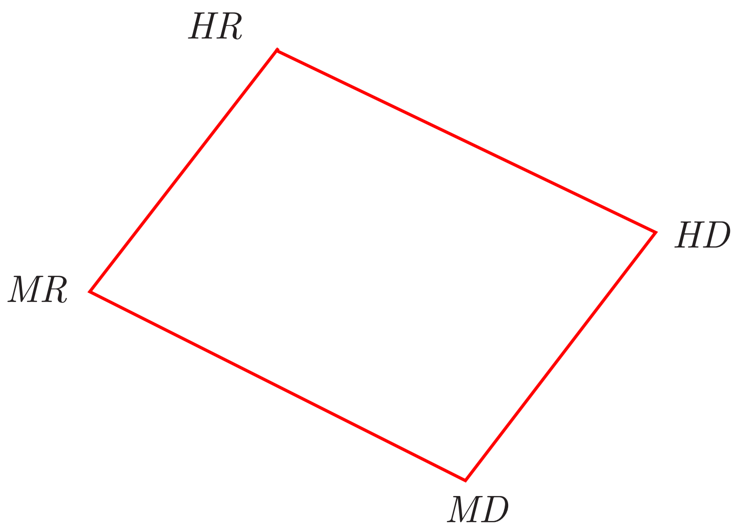
  where $\mathfrak{S}(\underbrace{s_0\, \alpha_0\, s_1\, \alpha_1 \ldots \alpha_{n-1}\, s_n}_{\text{history}})(\alpha) > 0$ implies $\alpha \in \mathit{Act}(s_n)$

  every D-policy is an R-policy; not the converse

- Thus: $MD \subset MR \subset HR$ and $MD \subset HD \subset HR$

Types of policies

- Distinguishing criteria:
  - Available information? current state (M), or history (H)
  - How to decide? deterministic (D) or randomized (R)
  - Fairness? (not today)
- The hierarchy of scheduler classes MD, MR, HD and HR:



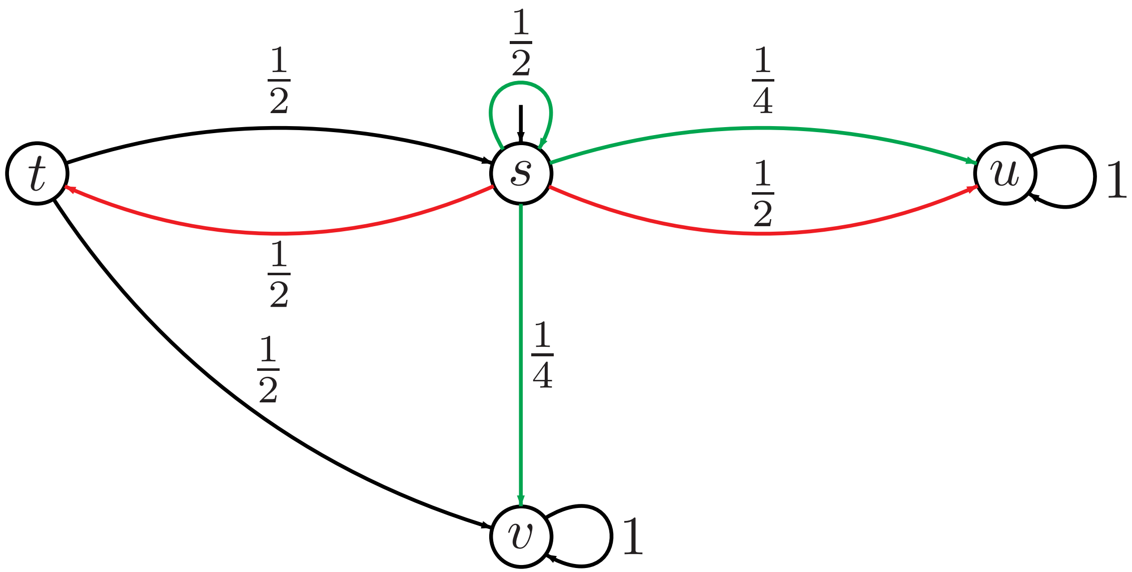
alternative terminology: tactic, scheduler, adversary, . . .

Applying a HD-Policy

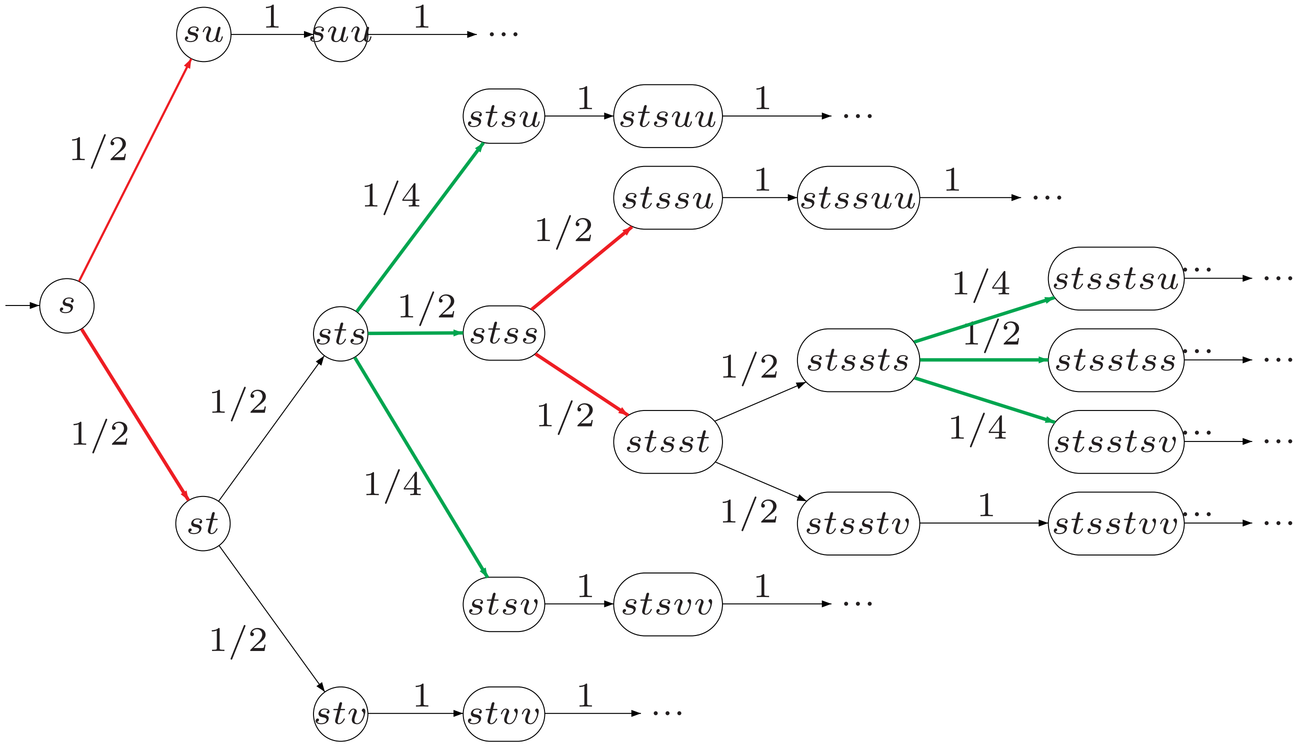
Policy $\mathfrak{S}$ on MDP $\mathcal{M} = (S, \mathit{Act}, \mathbf{P}, \mathit{AP}, L)$ with initial state s

- Basic idea: *unfold* $\mathcal{M}$, resolving the nondeterminism according to $\mathfrak{S}$
  - this yields a tree rooted at state s
- This yields the infinite Markov chain $\mathcal{M}_{\mathfrak{S}} = (S_{\mathfrak{S}}, \mathbf{P}_{\mathfrak{S}}, \iota_{init}, L_{\mathfrak{S}})$ with:
  - $S_{\mathfrak{S}} = S^+$, nonempty state sequences in MDP $\mathcal{M}$
  - $\mathbf{P}_{\mathfrak{S}}(\pi, \pi \rightarrow s) = \mathbf{P}(last(\pi), \mathfrak{S}(\pi), s)$ and 0 otherwise
  - $L_{\mathfrak{S}}(\pi) = L(last(\pi))$

Markov Decision Process



Applying a Policy



HD policy = alternate between red and green

Content of this lecture

- Markov decision processes
  - motivation, definition, policies

$\Rightarrow$ Reachability probabilities
  - quantitative and qualitative reachability

- Probabilistic CTL
  - syntax, semantics, model checking

Reachability Objectives in MDPs

- Reachability probability of set $B \subseteq S$ from state s:

$$\Pr^{\mathfrak{S}}(s \models \Diamond B) \;=\; \Pr_s^{\mathcal{M}_{\mathfrak{S}}}\{\, \pi \in \mathit{Paths}(s) \mid \pi \models \Diamond B \,\}$$

  - ω-regular properties (and many more) are also measurable
  - $\forall \mathfrak{S}.\ \Pr^{\mathfrak{S}}(s \models \Diamond B) \leqslant \varepsilon$ implies $\forall \mathfrak{S}.\ \Pr^{\mathfrak{S}}(s \models \Box \neg B) \geqslant 1 - \varepsilon$

Reachability Objectives in MDPs

- Reachability probability of set $B \subseteq S$ from state s:

$$\Pr^{\mathfrak{S}}(s \models \Diamond B) \;=\; \Pr_s^{\mathcal{M}_{\mathfrak{S}}}\{\, \pi \in \textit{Paths}(s) \mid \pi \models \Diamond B \,\}$$

  - ω-regular properties (and many more) are also measurable
  - $\forall \mathfrak{S}.\, \Pr^{\mathfrak{S}}(s \models \Diamond B) \leqslant \varepsilon$ implies $\forall \mathfrak{S}.\, \Pr^{\mathfrak{S}}(s \models \Box \neg B) \geqslant 1 - \varepsilon$

- Analysis focuses on obtaining lower- and upperbounds, e.g.,

$$\Pr^{\max}(s \models \Diamond B) \;=\; \sup_{\mathfrak{S}} \Pr^{\mathfrak{S}}(s \models \Diamond B)$$

  note: $\mathfrak{S}$ ranges over all, potentially infinitely many, policies

- And on determining policies (MD, HD, ...) for these bounds

Constrained reachability

- Let $B, C \subseteq S$ and consider the property $C \, \mathsf{U} \, B$ in MDP $\mathcal{M}$
- Remove all outgoing transitions from states in B, and $S \setminus (B \cup C)$
  - i.e., equip such state t with α_t with $\mathbf{P}(t, \alpha_t, t) = 1$
  - this yields the MDP $\mathcal{M}'$
- Then, it holds:

$$
\begin{aligned}
\Pr^{\max}_{\mathcal{M}}(s \models C \, \mathsf{U} \, B) &= \Pr^{\max}_{\mathcal{M}'}(s \models \Diamond B) \\
\Pr^{\min}_{\mathcal{M}}(s \models C \, \mathsf{U} \, B) &= \Pr^{\min}_{\mathcal{M}'}(s \models \Diamond B)
\end{aligned}
$$

$\Rightarrow$ constrained reachability objectives can be reduced to simple reachability

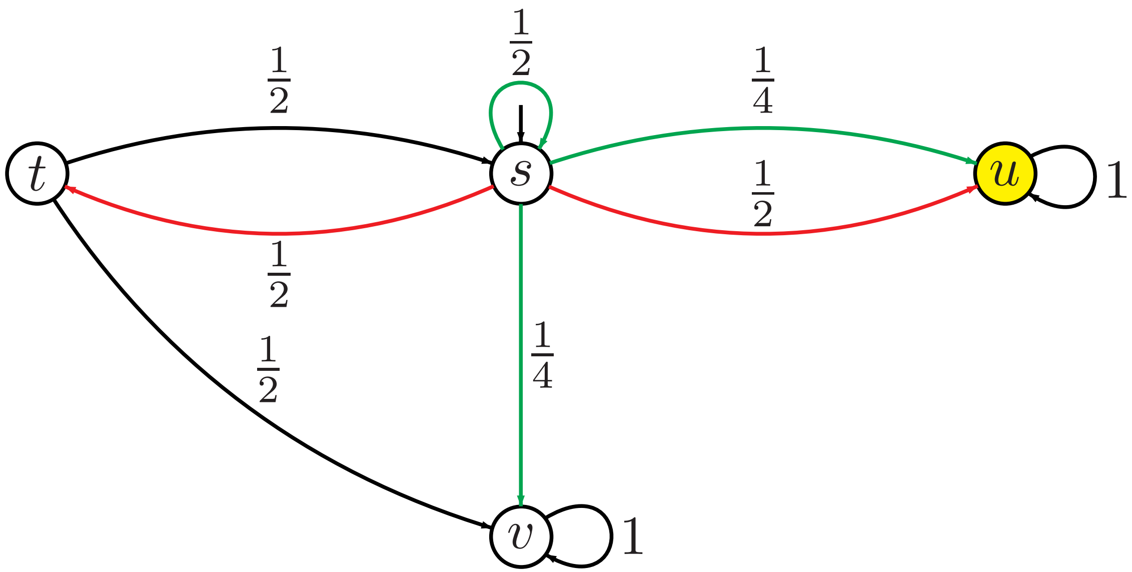
Reachability probabilities in finite MDPs

- Let variable $x_s = \Pr^{\max}(s \models \Diamond B)$ for any state s
- x_s is the unique solution of the set of equations:
  - if B is not reachable from s then $x_s = 0$
  - if $s \in B$ then $x_s = 1$
- For any state $s \in \mathit{Sat}(\exists\Diamond B) \setminus B$:

$$x_s \;=\; \max \left\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t \mid \alpha \in \mathit{Act}(s) \right\}$$

for minimal probabilities similar equations are obtained

Example



equation system for reachability objective $\Diamond\{\, u \,\}$ is:

$$x_u = 1 \text{ and } x_v = 0$$

$$x_s \;=\; \max\{\, \tfrac{1}{2}x_s + \tfrac{1}{4}x_u + \tfrac{1}{4}x_v,\ \tfrac{1}{2}x_u + \tfrac{1}{2}x_t \,\} \quad \text{and} \quad x_t = \tfrac{1}{2}x_s + \tfrac{1}{2}x_v$$

Reachability objectives

there exists an MD-policy $\mathfrak{S}$ with:

$$\Pr^{\mathfrak{S}}(s \models \Diamond B) = \Pr^{\max}(s \models \Diamond B)$$

- For $\Diamond^{\leqslant n} B$ with $n \in \mathbb{N}$, finite-memory policies are optimal
- Maximal reachability probabilities are obtained by a linear program
  - or, alternatively, by means of value iteration

$\Rightarrow$ Values $\Pr^{\max}(s \models \Diamond B)$ can be computed in polytime

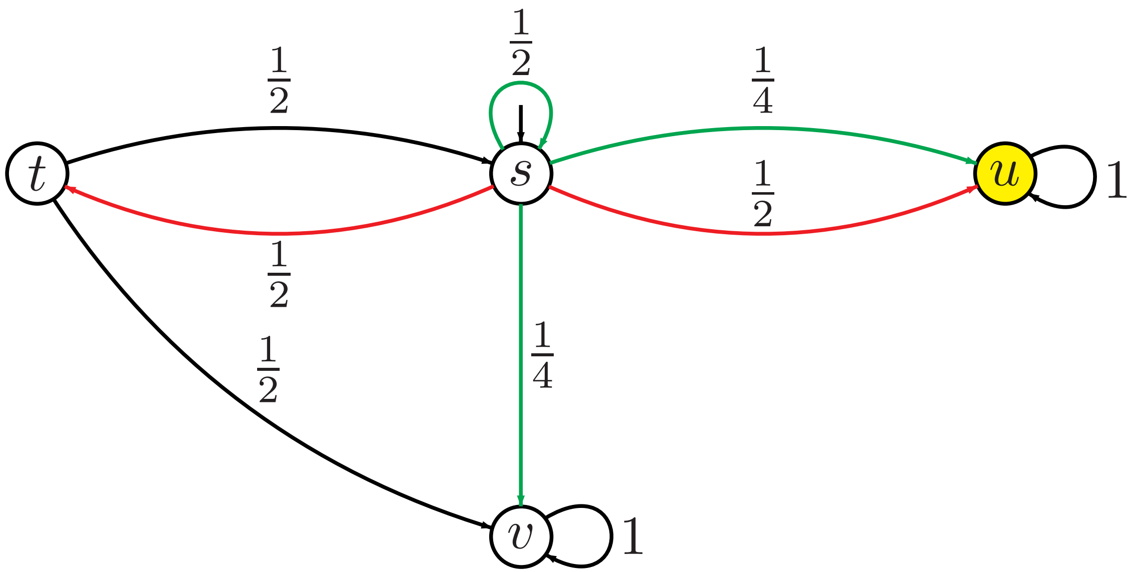
Linear program

- Let variable $x_s = \Pr^{\max}(s \models \Diamond B)$ for any state s
- x_s is the unique solution of the set of equations:
  - if $s \not\models \exists\Diamond B$ then $x_s = 0$
  - if $s \in B$ then $x_s = 1$
- For any state $s \in \mathit{Sat}(\exists\Diamond B) \setminus B$:

$$x_s \;\geqslant\; \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t \qquad \text{for any } \alpha \in \mathit{Act}(s)$$

- Such that $\sum_{s \in S} x_s$ is minimal

Example



LP problem for reachability objective $\Diamond\{\, u \,\}$ is:

minimize $\sum_{t \in S} x_t$ such that $x_u = 1$ and $x_v = 0$

$x_s \geqslant \frac{1}{2}x_s + \frac{1}{4}x_u + \frac{1}{4}x_v$ and $x_s \geqslant \frac{1}{2}x_u + \frac{1}{2}x_t$ and $x_t \geqslant \frac{1}{2}x_s + \frac{1}{2}x_v$

Asynchronous leader election

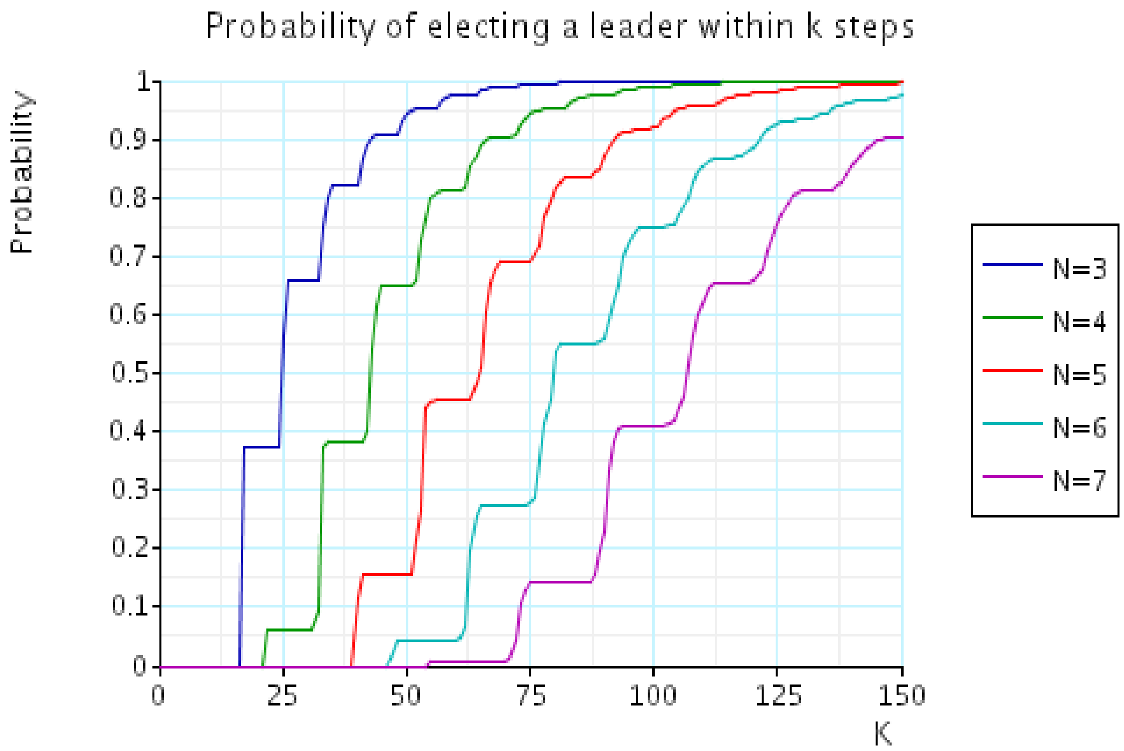
- An unidirectional asynchronous ring of $N > 2$ nodes
  - each process behaves asynchronously

  $\Rightarrow$ this interleaved concurrency gives rise to an MDP!

- Each node is initially active and proceeds as follows:
  - flip a fair coin (0 and 1), and pass the outcome to your right neighbour
  - if you have chosen 0 while your left neighbour has passed 1, become inactive
  - send a counter around the ring: if only active node $\Rightarrow$ become leader

(Itai & Rodeh, 1990)

Probability to elect a leader within k steps



$\mathbb{P}_{\leqslant q}(\Diamond^{\leqslant k}$ *leader elected*$)$ © PRISM web-page

maximum and minimum probabilities coincide in this case

Content of this lecture

- Markov decision processes
  - motivation, definition, policies
- Reachability probabilities
  - quantitative and qualitative reachability

$\Rightarrow$ Probabilistic CTL
  - syntax, semantics, model checking

PCTL Syntax

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

$$\boxed{\begin{array}{rcl} \Phi & ::= & \text{true} \;\Big|\; a \;\Big|\; \Phi \wedge \Phi \;\Big|\; \neg\Phi \;\Big|\; \mathbb{P}_J(\varphi) \\[2ex] \varphi & ::= & \bigcirc\Phi \;\Big|\; \Phi_1 \,\mathsf{U}\, \Phi_2 \;\Big|\; \Phi_1 \,\mathsf{U}^{\leqslant n}\, \Phi_2 \end{array}}$$

- $s_0\alpha_0s_1\alpha_1s_2\ldots \models \Phi \,\mathsf{U}^{\leqslant n}\, \Psi$ 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 for all policies

Derived operators

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

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

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

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

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

PCTL semantics (1)

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

Relation $\models$ is defined by:

$$
\begin{array}{lll}
s \models a & \text{iff} & a \in L(s) \\
s \models \neg \Phi & \text{iff} & \text{not } (s \models \Phi) \\
s \models \Phi \vee \Psi & \text{iff} & (s \models \Phi) \text{ or } (s \models \Psi) \\
s \models \mathbb{P}_J(\varphi) & \text{iff} & \Pr^{\mathfrak{S}}(s \models \varphi) \in J \text{ for all policies } \mathfrak{S}
\end{array}
$$

where $\Pr^{\mathfrak{S}}(s \models \varphi) = \Pr^{\mathfrak{S}}_s\{\pi \in \mathit{Paths}(s) \mid \pi \models \varphi\}$

Remarks

$$s \models \mathbb{P}_{J}(\varphi) \quad \text{iff} \quad \Pr^{\mathfrak{S}}(s \models \varphi) \in J \text{ for all policies } \mathfrak{S}$$

so:

$$s \models \mathbb{P}_{\leqslant p}(\varphi) \quad \text{iff} \quad \Pr^{\max}(s \models \varphi) \leqslant p$$

$$s \models \mathbb{P}_{\geqslant p}(\varphi) \quad \text{iff} \quad \Pr^{\min}(s \models \varphi) \geqslant p$$

note that: $\mathbb{P}_{\leqslant p}(\varphi) \quad \not\equiv \quad \neg\mathbb{P}_{>p}(\varphi)$

PCTL semantics (2)

A *path* is an infinite sequence $s_0 \xrightarrow{\alpha_0} s_1 \xrightarrow{\alpha_1} \ldots$ with $\mathbf{P}(s_i, \alpha_i, s_{i+1}) > 0$

Semantics of path-formulas is defined as for DTMCs:

$$
\begin{array}{lll}
\pi \models \bigcirc \Phi & \text{iff} & s_1 \models \Phi \\
\pi \models \Phi \,\mathsf{U}\, \Psi & \text{iff} & \exists n \geqslant 0. (s_n \models \Psi \wedge \forall 0 \leqslant i < n.\, s_i \models \Phi) \\
\pi \models \Phi \,\mathsf{U}^{\leqslant n}\, \Psi & \text{iff} & \exists k \geqslant 0. (k \leqslant n \wedge s_k \models \Psi \wedge \\
 & & \qquad \forall 0 \leqslant i < k.\, s_i \models \Phi)
\end{array}
$$

PCTL model checking

- Given a finite MDP $\mathcal{M}$ and PCTL formula Φ, how to check $\mathcal{M} \models \Phi$?
- Check whether state s in a MDP 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

- $s \models \mathbb{P}_J(\Phi \, \mathsf{U} \, \Psi)$ if and only if $\Pr^{\max}(s \models \Phi \, \mathsf{U} \, \Psi) \; \in \; J$
- $\Pr(s \models \Phi \, \mathsf{U} \, \Psi)$ is the unique solution of: (Bianco & de Alfaro, 1998)
  - 1 if $s \models \Psi$
  - for $s \models \Phi \wedge \neg \Psi$:

$$\max_{\alpha} \left\{ \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot \Pr(s' \models \Phi \, \mathsf{U} \, \Psi) \right\}$$

  - 0 otherwise
- Possible efficiency improvement by graph-theoretical pre-computation

Time complexity

For finite MDP $\mathcal{M}$ and PCTL formula Φ, $\mathcal{M} \models \Phi$ can be solved in time

$$\mathcal{O}\big(\, poly(|\mathcal{M}|) \,\cdot\, n_{\max} \cdot |\Phi| \,\big)$$

where $n_{\max} \;=\; \max\{\, n \mid \Psi_1 \,\mathsf{U}^{\leqslant n}\, \Psi_2$ occurs in $\Phi \,\}$ with $\max \varnothing = 1$

Extensions

- LTL model checking
- Costs
- Abstraction
  - bisimulation minimization, partial-order reduction, MTBDDs, . . .
- Continuous time
- Fairness

Probabilistic model checking

- . . . . . . is a mature automated technique
- . . . . . . has a broad range of applications
- . . . . . . is supported by powerful software tools
- . . . . . . recent significant efficiency gain
- . . . . . . offers many interesting challenges!

more information: `moves.rwth-aachen.de/˜katoen`