Probabilistic Model Checking

Marta Kwiatkowska
Dave Parker

Oxford University Computing Laboratory

ESSLLI'10 Summer School, Copenhagen, August 2010

Course overview

- 5 lectures: Mon–Fri, 11am–12.30pm
  - Introduction
  - 1 – Discrete time Markov chains
  - 2 – Markov decision processes
  - 3 – Continuous-time Markov chains
  - 4 – Probabilistic model checking in practice
  - 5 – Probabilistic timed automata
- Course materials available here:
  - http://www.prismmodelchecker.org/lectures/esslli10/
  - lecture slides, reference list

Part 2

Markov decision processes

Other probabilistic models

- What's missing from DTMCs?
- Nondeterminism
  - Markov decision processes (MDPs)…
- Real-time
  - continuous-time Markov chains (CTMCs)
    - exponentially distributed delays
  - probabilistic timed automata (PTAs)
    - real-valued clocks, discrete probabilistic choice, nondeterminism

Overview (Part 2)

- Markov decision processes (MDPs)
- Adversaries & probability spaces
- PCTL for MDPs
- PCTL model checking
- Further model checking (LTL, costs & rewards)
- Case study: Firewire root contention

Nondeterminism

- Some aspects of a system may not be probabilistic and should not be modelled probabilistically; for example:

- Concurrency – scheduling of parallel components
  - e.g. randomised distributed algorithms – multiple probabilistic processes operating asynchronously

- Underspecification – unknown model parameters
  - e.g. a probabilistic communication protocol designed for message propagation delays of between d_{min} and d_{max}

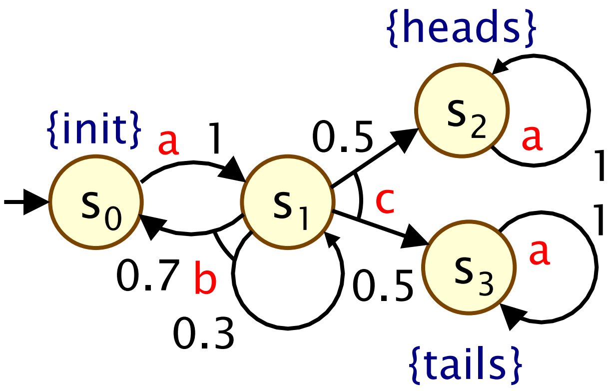
- Unknown environments
  - e.g. probabilistic security protocols – unknown adversary

Markov decision processes

- Markov decision processes (MDPs)
  - extension of DTMCs which allow nondeterministic choice

- Like DTMCs:
  - discrete set of states representing possible configurations of the system being modelled
  - transitions between states occur in discrete time-steps

- Probabilities and nondeterminism
  - in each state, a nondeterministic choice between several discrete probability distributions over successor states

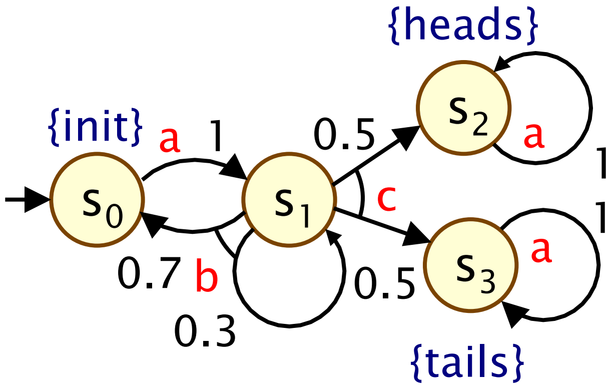


Markov decision processes

- Formally, an MDP M is a tuple $(S, s_{init}, \mathbf{Steps}, L)$ where:
  - S is a finite set of states ("state space")
  - $s_{init} \in S$ is the initial state
  - **Steps** : $S \rightarrow 2^{Act \times Dist(S)}$ is the transition probability function

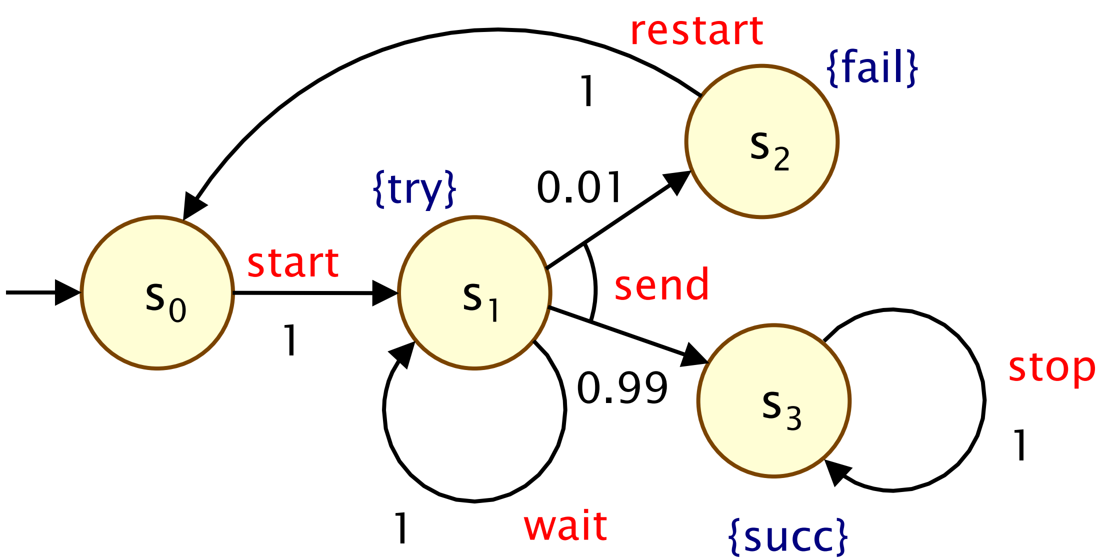
    where Act is a set of actions and Dist(S) is the set of discrete probability distributions over the set S
  - $L : S \rightarrow 2^{AP}$ is a labelling with atomic propositions

- Notes:
  - **Steps**(s) is always non-empty, i.e. no deadlocks
  - the use of actions to label distributions is optional



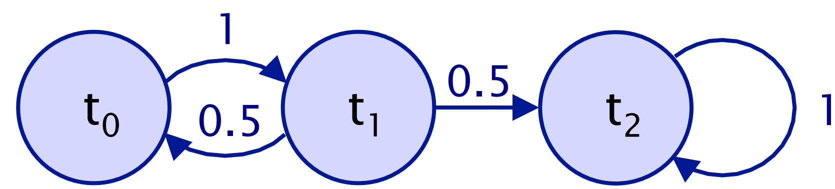
Simple MDP example

- Modification of the simple DTMC communication protocol
  - after one step, process starts trying to send a message
  - then, a nondeterministic choice between: (a) waiting a step because the channel is unready; (b) sending the message
  - if the latter, with probability 0.99 send successfully and stop
  - and with probability 0.01, message sending fails, restart

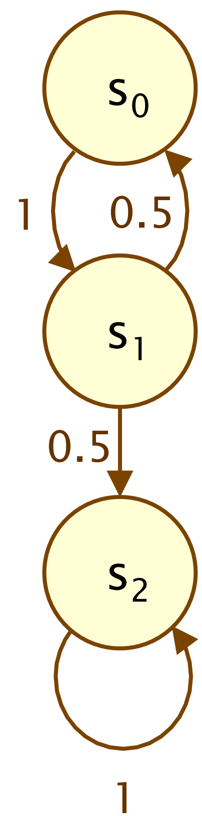


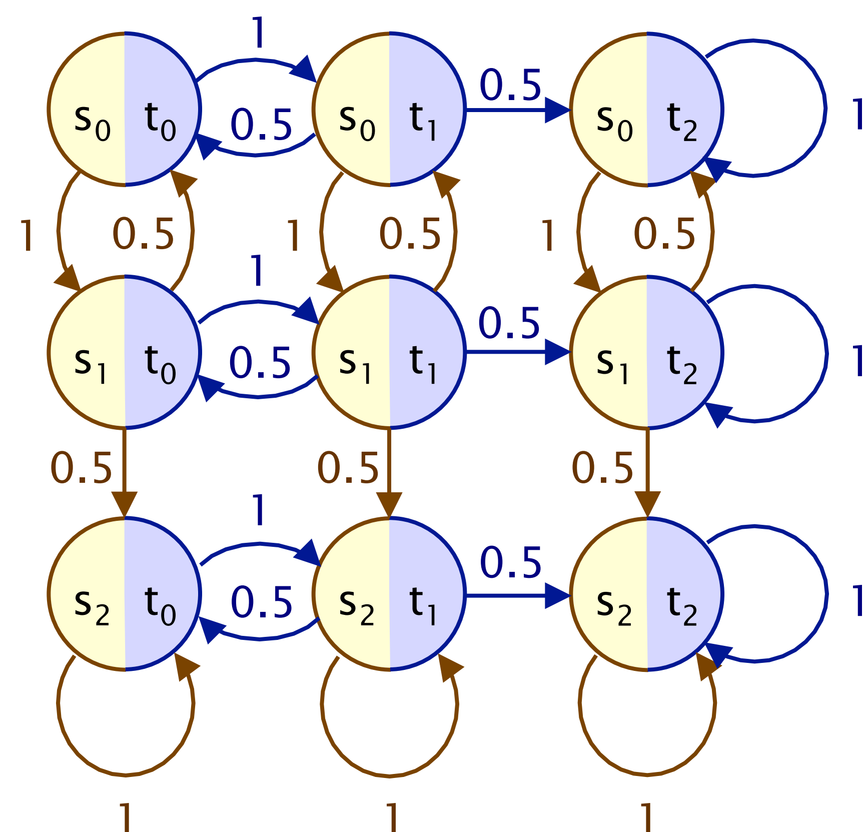
Example – Parallel composition

Asynchronous parallel composition of two 3-state DTMCs



Action labels omitted here





Paths and probabilities

- A (finite or infinite) path through an MDP
  - is a sequence of states and action/distribution pairs
  - e.g. $s_0(a_0,\mu_0)s_1(a_1,\mu_1)s_2\ldots$
  - such that $(a_i,\mu_i) \in$ **Steps**(s_i) and $\mu_i(s_{i+1}) > 0$ for all $i \geq 0$
  - represents an execution (i.e. one possible behaviour) of the system which the MDP is modelling
  - note that a path resolves both types of choices: nondeterministic and probabilistic

- To consider the probability of some behaviour of the MDP
  - first need to resolve the nondeterministic choices
  - …which results in a DTMC
  - …for which we can define a probability measure over paths

Overview (Part 2)

- Markov decision processes (MDPs)
- Adversaries & probability spaces
- PCTL for MDPs
- PCTL model checking
- Further model checking (LTL, costs & rewards)
- Case study: Firewire root contention

Adversaries

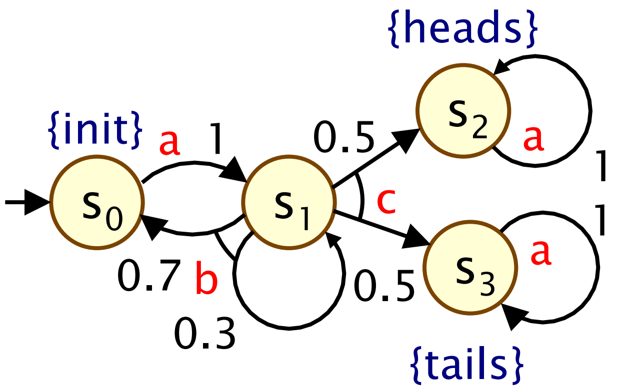
- An adversary resolves nondeterministic choice in an MDP
  - also known as "schedulers", "strategies" or "policies"

- Formally:
  - an adversary A of an MDP M is a function mapping every finite path $\omega = s_0(a_1,\mu_1)s_1 \ldots s_n$ to an element of **Steps**(s_n)

- For each A can define a probability measure Pr^A_s over paths
  - constructed through an infinite state DTMC $(Path^A_{fin}(s), s, \mathbf{P}^A_s)$
  - states of the DTMC are the finite paths of A starting in state s
  - initial state is s (the path starting in s of length 0)
  - $\mathbf{P}^A_s(\omega,\omega') = \mu(s)$ if $\omega' = \omega(a, \mu)s$ and $A(\omega) = (a,\mu)$
  - $\mathbf{P}^A_s(\omega,\omega') = 0$ otherwise

Adversaries – Examples

- Consider the simple MDP below
  - note that s_1 is the only state for which $|\mathbf{Steps}(s)| > 1$
  - i.e. s_1 is the only state for which an adversary makes a choice
  - let μ_b and μ_c denote the probability distributions associated with actions b and c in state s_1

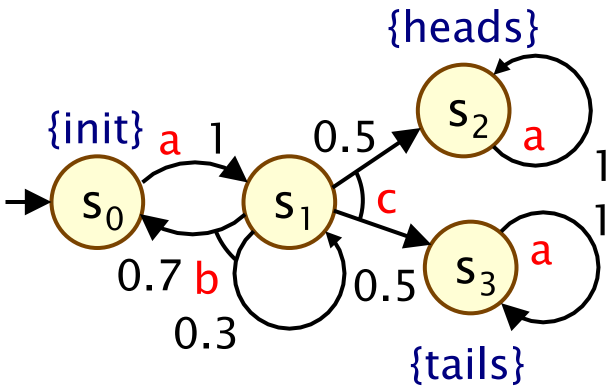


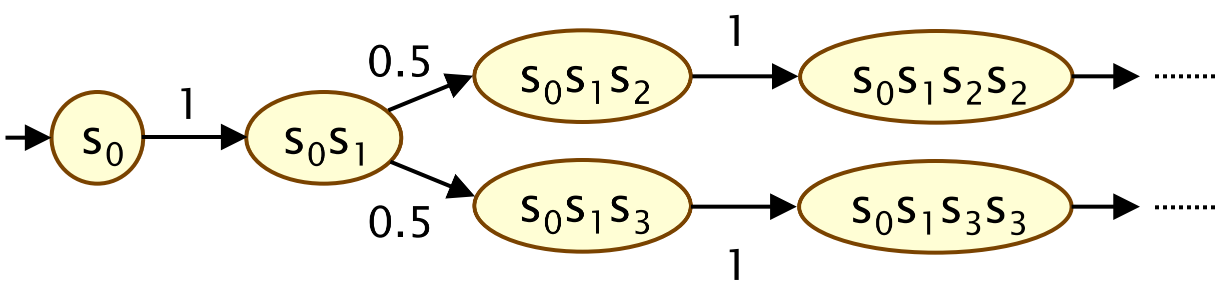
- Adversary A_1
  - picks action c the first time
  - $A_1(s_0s_1)=(c,\mu_c)$

- Adversary A_2
  - picks action b the first time, then c
  - $A_2(s_0s_1)=(b,\mu_b)$, $A_2(s_0s_1s_1)=(c,\mu_c)$, $A_2(s_0s_1s_0s_1)=(c,\mu_c)$

Adversaries – Examples

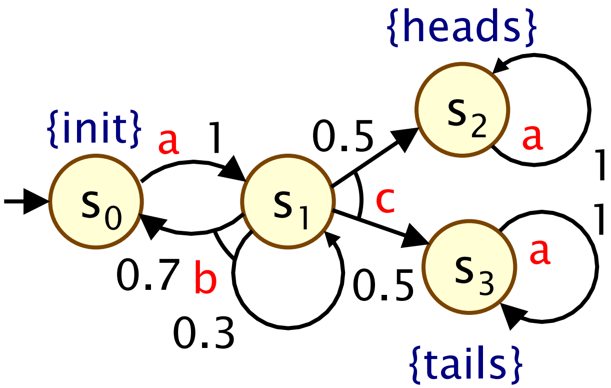
- Fragment of DTMC for adversary A_1
  - A_1 picks action c the first time

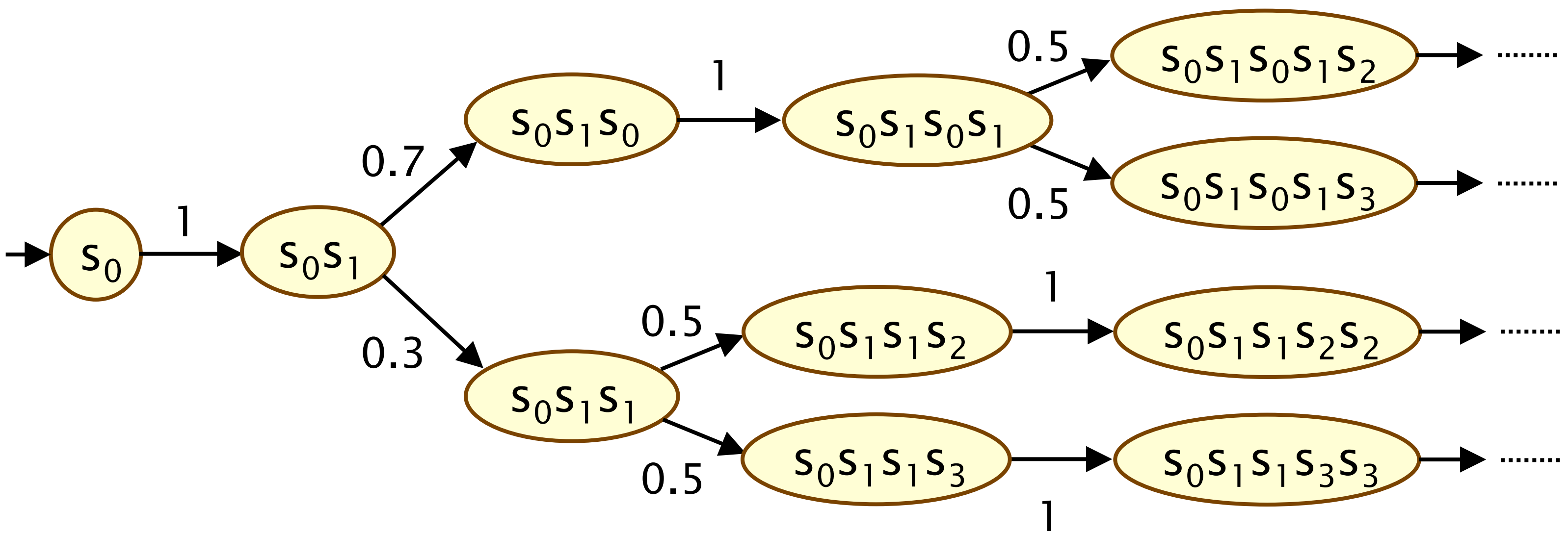




Adversaries – Examples

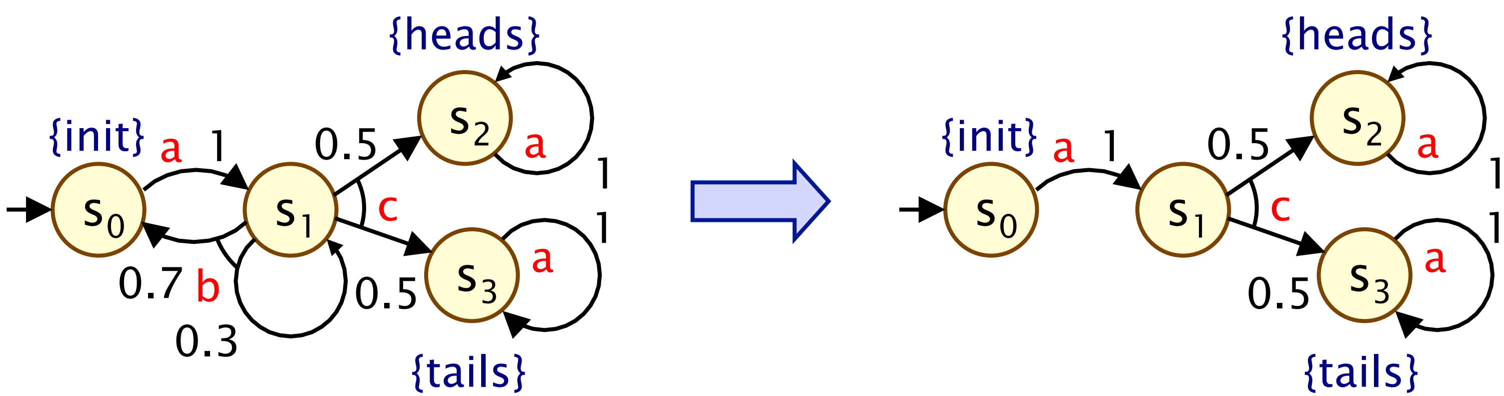
- Fragment of DTMC for adversary A_2
  - A_2 picks action b, then c





Memoryless adversaries

- Memoryless adversaries always pick same choice in a state
  - also known as: positional, Markov, simple
  - formally, for adversary A:
  - $A(s_0(a_1,\mu_1)s_1\ldots s_n)$ depends only on s_n
  - resulting DTMC can be mapped to a |S|-state DTMC
- From previous example:
  - adversary A_1 (picks c in s_1) is memoryless, A_2 is not



Overview (Part 2)

- Markov decision processes (MDPs)
- Adversaries & probability spaces
- PCTL for MDPs
- PCTL model checking
- Further model checking (LTL, costs & rewards)
- Case study: Firewire root contention

PCTL for MDPs

- The temporal logic PCTL can also describe MDP properties
- Identical syntax to the DTMC case:

ψ is true with probability ~p

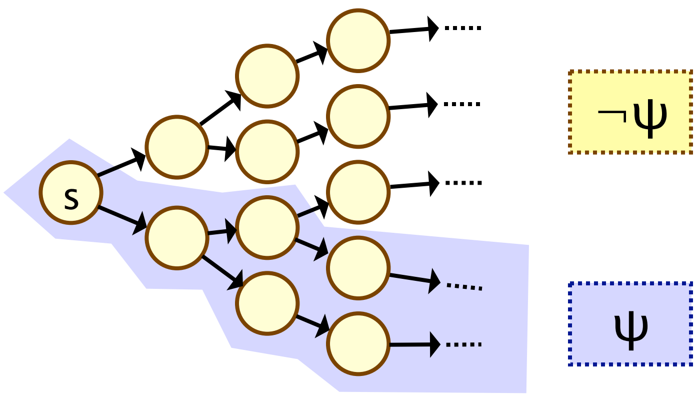
  - $\phi ::= \text{true} \mid a \mid \phi \wedge \phi \mid \neg\phi \mid P_{\sim p}\,[\,\psi\,]$ (state formulas)
  - $\psi ::= X\,\phi \mid \phi\ U^{\leq k}\ \phi \mid \phi\ U\ \phi$ (path formulas)

"next" "bounded until" "until"

- Semantics are also the same as DTMCs for:
  - atomic propositions, logical operators, path formulas

PCTL semantics for MDPs

- Semantics of the probabilistic operator P
  - can only define probabilities for a specific adversary A
  - $s \vDash P_{\sim p} [\ \psi\ ]$ means "the probability, from state s, that ψ is true for an outgoing path satisfies $\sim p$ for all adversaries A"
  - formally $s \vDash P_{\sim p} [\ \psi\ ] \Leftrightarrow Prob^A(s, \psi) \sim p$ for all adversaries A
  - where $Prob^A(s, \psi) = Pr^A_s \{ \omega \in Path^A(s) \mid \omega \vDash \psi \}$



$Prob^A(s, \psi) \sim p$

Minimum and maximum probabilities

- Letting:
  - $p_{max}(s, \psi) = \sup_A \text{Prob}^A(s, \psi)$
  - $p_{min}(s, \psi) = \inf_A \text{Prob}^A(s, \psi)$
- We have:
  - if $\sim \in \{\geq, >\}$, then $s \models P_{\sim p} [\ \psi\ ] \quad \Leftrightarrow \quad p_{min}(s, \psi) \sim p$
  - if $\sim \in \{<, \leq\}$, then $s \models P_{\sim p} [\ \psi\ ] \quad \Leftrightarrow \quad p_{max}(s, \psi) \sim p$
- Model checking $P_{\sim p}[\ \psi\ ]$ reduces to the computation over all adversaries of either:
  - the minimum probability of ψ holding
  - the maximum probability of ψ holding
- Crucial result for model checking PCTL on MDPs
  - memoryless adversaries suffice, i.e. there are always memoryless adversaries A_{min} and A_{max} for which:
  - $\text{Prob}^{Amin}(s, \psi) = p_{min}(s, \psi)$ and $\text{Prob}^{Amax}(s, \psi) = p_{max}(s, \psi)$

Quantitative properties

- For PCTL properties with P as the outermost operator
  - quantitative form (two types): $Pmin_{=?}$ [ψ] and $Pmax_{=?}$ [ψ]
  - i.e. "what is the minimum/maximum probability (over all adversaries) that path formula ψ is true?"
  - corresponds to an analysis of best-case or worst-case behaviour of the system
  - model checking is no harder since compute the values of $p_{min}(s, \psi)$ or $p_{max}(s, \psi)$ anyway
  - useful to spot patterns/trends

- Example: CSMA/CD protocol
  - "min/max probability
    that a message is sent
    within the deadline"

Other classes of adversary

- A more general semantics for PCTL over MDPs
  - parameterise by a class of adversaries Adv
- Only change is:
  - $s \vDash_{Adv} P_{\sim p} [\psi] \Leftrightarrow Prob^A(s, \psi) \sim p$ for all adversaries $A \in Adv$
- Original semantics obtained by taking Adv to be the set of all adversaries for the MDP
- Alternatively, take Adv to be the set of all fair adversaries
  - path fairness: if a state is occurs on a path infinitely often, then each non-deterministic choice occurs infinite often
  - see e.g. [BK98]

Some real PCTL examples

- Byzantine agreement protocol
  - $\mathrm{Pmin}_{=?}$ [F (agreement $\wedge$ rounds$\leq$2)]
  - "what is the minimum probability that agreement is reached within two rounds?"
- CSMA/CD communication protocol
  - $\mathrm{Pmax}_{=?}$ [F collisions=k]
  - "what is the maximum probability of k collisions?"
- Self-stabilisation protocols
  - $\mathrm{Pmin}_{=?}$ [$F^{\leq t}$ stable]
  - "what is the minimum probability of reaching a stable state within k steps?"

PCTL model checking for MDPs

- Algorithm for PCTL model checking [BdA95]
  - inputs: MDP $M=(S,s_{init},\mathbf{Steps},L)$, PCTL formula ϕ
  - output: $Sat(\phi) = \{ s \in S \mid s \vDash \phi \}$ = set of states satisfying ϕ

- Basic algorithm same as PCTL model checking for DTMCs
  - proceeds by induction on parse tree of ϕ
  - non-probabilistic operators (true, a, $\neg$, $\wedge$) straightforward

- Only need to consider $P_{\sim p}$ [ψ] formulas
  - reduces to computation of $p_{min}(s, \psi)$ or $p_{max}(s, \psi)$ for all $s \in S$
  - dependent on whether $\sim \in \{\geq,>\}$ or $\sim \in \{<,\leq\}$
  - these slides cover the case $p_{min}(s, \phi_1\ U\ \phi_2)$, i.e. $\sim \in \{\geq,>\}$
  - case for maximum probabilities is very similar
  - next ($X\ \phi$) and bounded until ($\phi_1\ U^{\leq k}\ \phi_2$) are straightforward extensions of the DTMC case

Overview (Part 2)

- Markov decision processes (MDPs)
- Adversaries & probability spaces
- PCTL for MDPs
- PCTL model checking
- Further model checking (LTL, costs & rewards)
- Case study: Firewire root contention

PCTL until for MDPs

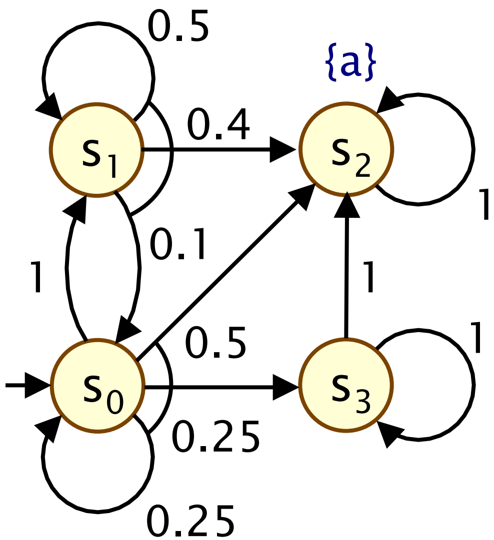
- Computation of probabilities $p_{min}(s, \phi_1 \text{ U } \phi_2)$ for all $s \in S$
- First identify all states where the probability is 1 or 0
  - "precomputation" algorithms, yielding sets S^{yes}, S^{no}
- Then compute (min) probabilities for remaining states ($S^?$)
  - either: solve linear programming problem
  - or: approximate with an iterative solution method
  - or: use policy iteration

Example:

$P_{\geq p}$ [F a]

$\equiv$

$P_{\geq p}$ [true U a]

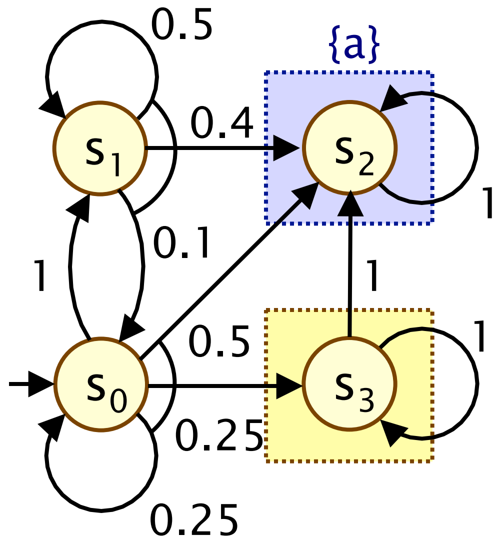


PCTL until – Precomputation

- Identify all states where $p_{min}(s, \phi_1 \text{ U } \phi_2)$ is 1 or 0
  - S^{yes} = $\text{Sat}(P_{\geq 1} [\ \phi_1 \text{ U } \phi_2\ ])$, S^{no} = $\text{Sat}(\neg\ P_{>0} [\ \phi_1 \text{ U } \phi_2\ ])$
- Two graph-based precomputation algorithms:
  - algorithm Prob1A computes S^{yes}
    - for all adversaries the probability of satisfying $\phi_1 \text{ U } \phi_2$ is 1
  - algorithm Prob0E computes S^{no}
    - there exists an adversary for which the probability is 0

Example:

$P_{\geq p}$ [F a]



S^{yes} = $\text{Sat}(P_{\geq 1}$ [F a])

S^{no} = $\text{Sat}(\neg P_{>0}$ [F a])

Method 1 – Linear programming

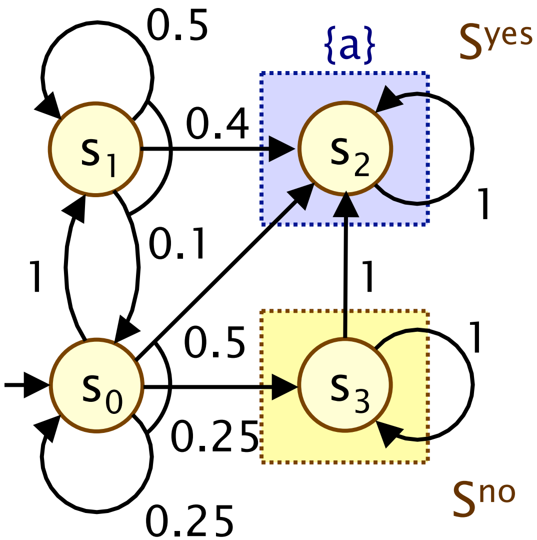
- Probabilities $p_{min}(s, \phi_1 \text{ U } \phi_2)$ for remaining states in the set $S^? = S \setminus (S^{yes} \cup S^{no})$ can be obtained as the unique solution of the following linear programming (LP) problem:

$$\text{maximize } \sum_{s \in S^?} x_s \text{ subject to the constraints:}$$
$$x_s \leq \sum_{s' \in S^?} \mu(s') \cdot x_{s'} + \sum_{s' \in S^{yes}} \mu(s')$$
$$\text{for all } s \in S^? \text{ and for all } (a,\mu) \in Steps(s)$$

- Simple case of a more general problem known as the stochastic shortest path problem [BT91]

- This can be solved with standard techniques
  - e.g. Simplex, ellipsoid method, branch-and-cut

Example – PCTL until (LP)



Let $x_i = p_{min}(s_i, F\ a)$

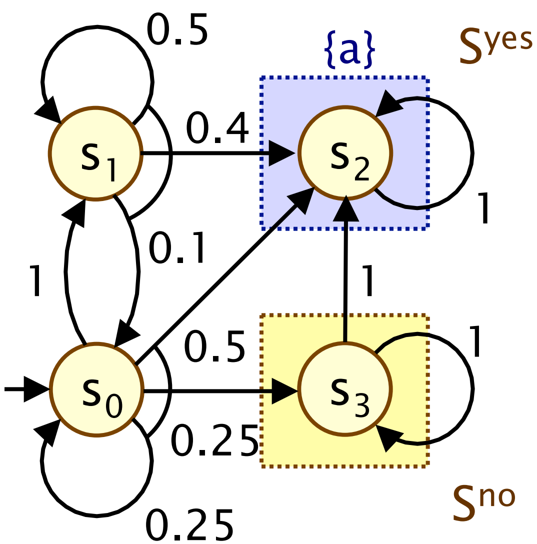
S^{yes}: $x_2=1$, S^{no}: $x_3=0$

For $S^? = \{x_0, x_1\}$:

Maximise x_0+x_1 subject to constraints:

- $x_0 \leq x_1$
- $x_0 \leq 0.25 \cdot x_0 + 0.5$
- $x_1 \leq 0.1 \cdot x_0 + 0.5 \cdot x_1 + 0.4$

Example – PCTL until (LP)



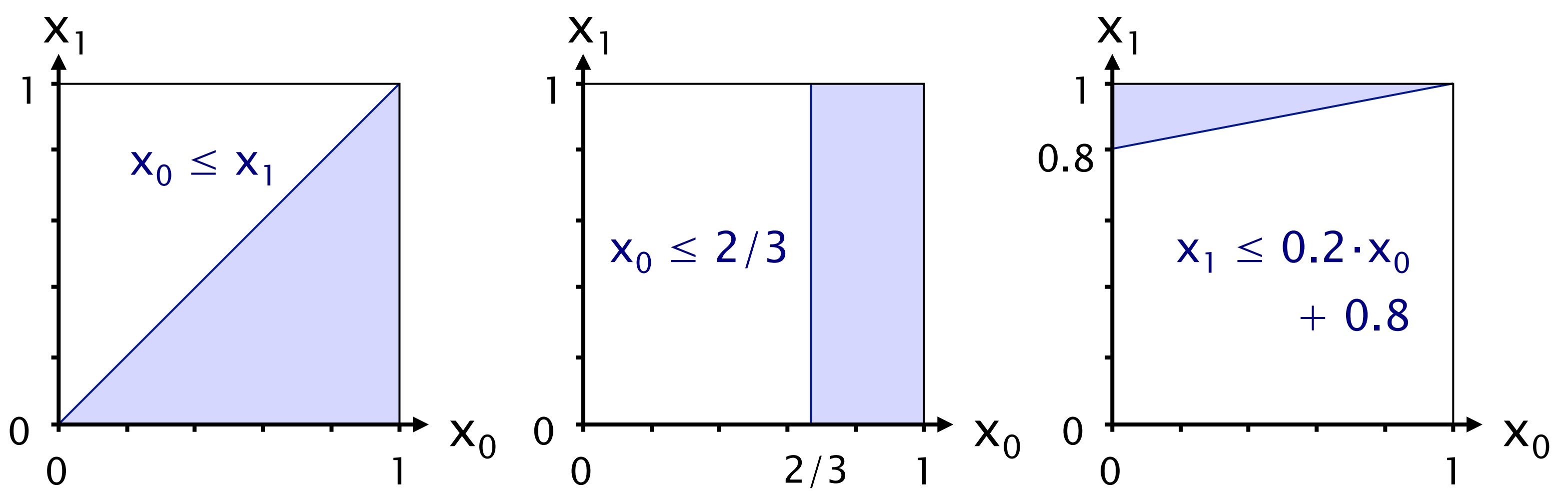
Let $x_i = p_{min}(s_i, F\ a)$

S^{yes}: $x_2=1$, S^{no}: $x_3=0$

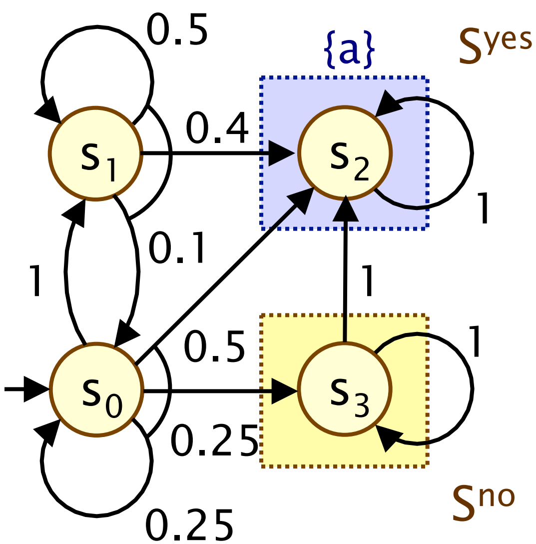
For $S^? = \{x_0, x_1\}$:

Maximise x_0+x_1 subject to constraints:

- $x_0 \leq x_1$
- $x_0 \leq 2/3$
- $x_1 \leq 0.2 \cdot x_0 + 0.8$



Example – PCTL until (LP)



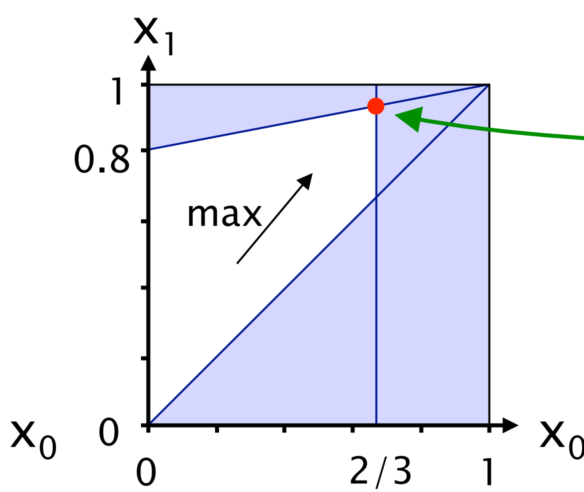
Let $x_i = p_{min}(s_i, F\ a)$

S^{yes}: $x_2=1$, S^{no}: $x_3=0$

For $S^? = \{x_0, x_1\}$:

Maximise x_0+x_1 subject to constraints:

- $x_0 \leq x_1$
- $x_0 \leq 2/3$
- $x_1 \leq 0.2 \cdot x_0 + 0.8$



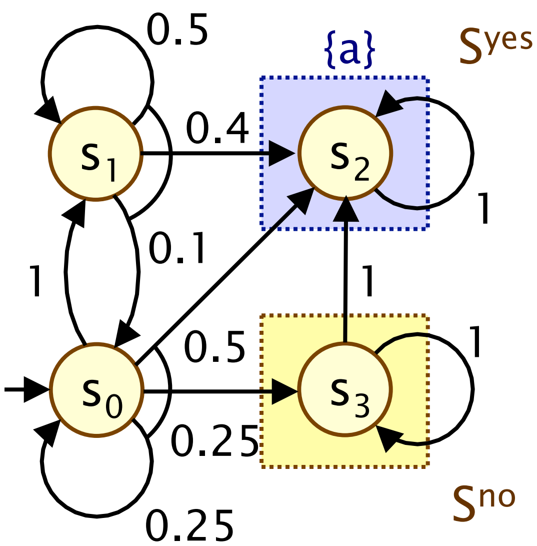
Solution:

(x_0, x_1)

$=$

$(2/3, 14/15)$

Example – PCTL until (LP)



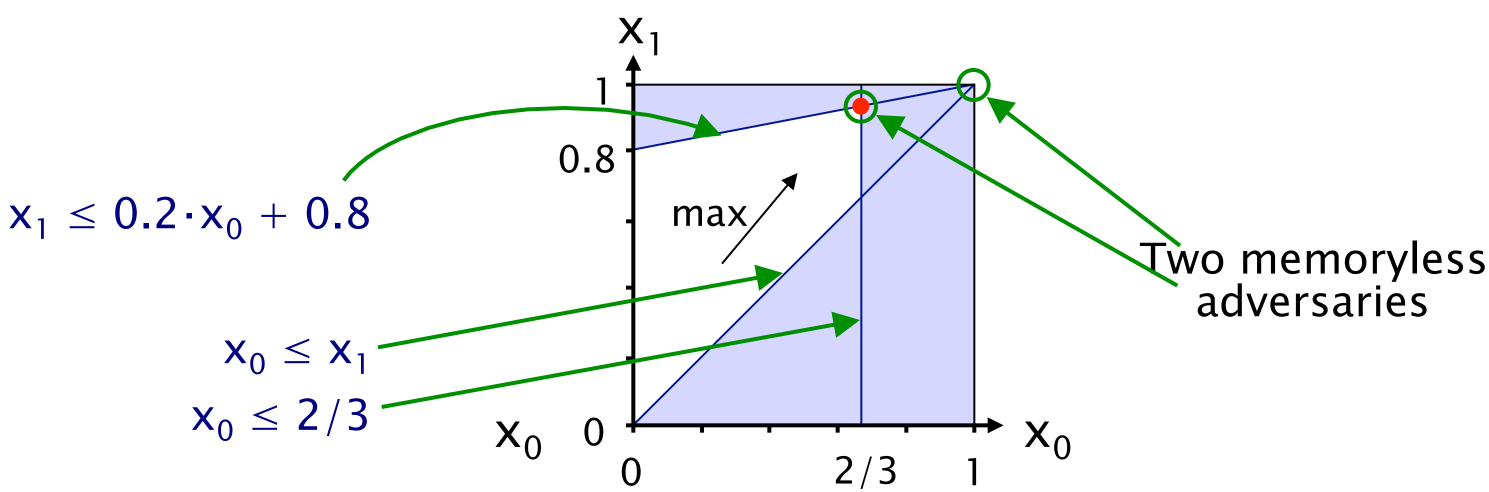
Let $x_i = p_{min}(s_i, \text{F } a)$

S^{yes}: $x_2=1$, S^{no}: $x_3=0$

For $S^? = \{x_0, x_1\}$:

Maximise x_0+x_1 subject to constraints:

- $x_0 \leq x_1$
- $x_0 \leq 2/3$
- $x_1 \leq 0.2 \cdot x_0 + 0.8$



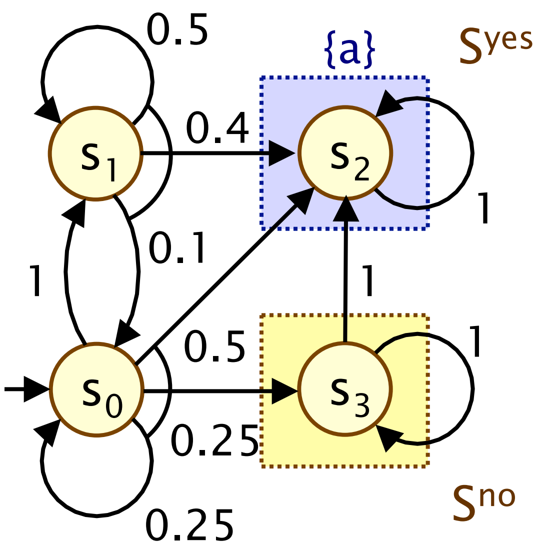
Method 2 – Value iteration

- For probabilities $p_{min}(s, \phi_1 \text{ U } \phi_2)$ it can be shown that:
  - $p_{min}(s, \phi_1 \text{ U } \phi_2) = \lim_{n\to\infty} x_s^{(n)}$ where:

$$
x_s^{(n)} = \begin{cases} 1 & \text{if } s \in S^{yes} \\ 0 & \text{if } s \in S^{no} \\ 0 & \text{if } s \in S^{?} \text{ and } n = 0 \\ \min_{(a,\mu)\in Steps(s)} \left(\sum_{s' \in S} \mu(s') \cdot x_{s'}^{(n-1)} \right) & \text{if } s \in S^{?} \text{ and } n > 0 \end{cases}
$$

- This forms the basis for an (approximate) iterative solution
  - iterations terminated when solution converges sufficiently

Example – PCTL until (value iteration)



Compute: $p_{min}(s_i, F\ a)$

$S^{yes} = \{x_2\}$, $S^{no} = \{x_3\}$, $S^{?} = \{x_0, x_1\}$

$[\ x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}\ ]$

n=0: $[\ 0, 0, 1, 0\ ]$

n=1: $[\ \min(0, 0.25 \cdot 0 + 0.5),$

$0.1 \cdot 0 + 0.5 \cdot 0 + 0.4,\ 1,\ 0\ ]$

$= [\ 0, 0.4, 1, 0\ ]$

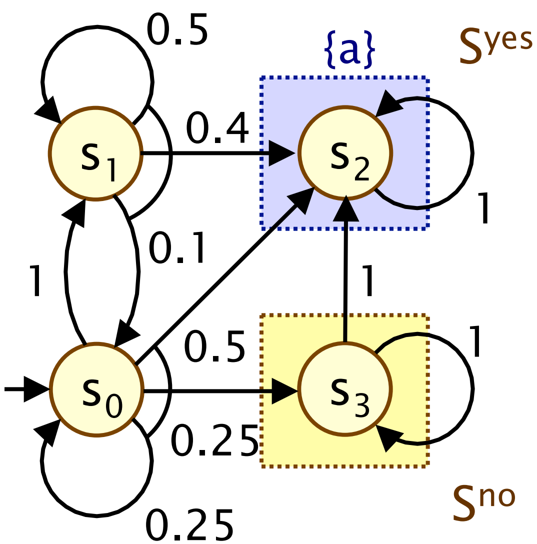
n=2: $[\ \min(0.4, 0.25 \cdot 0 + 0.5),$

$0.1 \cdot 0 + 0.5 \cdot 0.4 + 0.4,\ 1,\ 0\ ]$

$= [\ 0.4, 0.6, 1, 0\ ]$

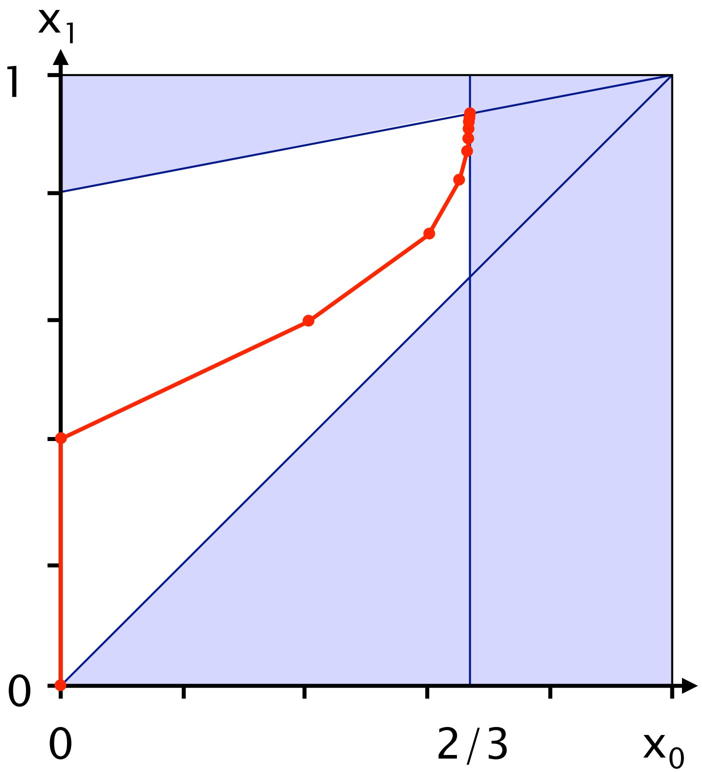
n=3: …

Example – PCTL until (value iteration)



	$[\ x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}\ ]$
n=0:	[0.000000, 0.000000, 1, 0]
n=1:	[0.000000, 0.400000, 1, 0]
n=2:	[0.400000, 0.600000, 1, 0]
n=3:	[0.600000, 0.740000, 1, 0]
n=4:	[0.650000, 0.830000, 1, 0]
n=5:	[0.662500, 0.880000, 1, 0]
n=6:	[0.665625, 0.906250, 1, 0]
n=7:	[0.666406, 0.919688, 1, 0]
n=8:	[0.666602, 0.926484, 1, 0]
n=9:	[0.666650, 0.929902, 1, 0]
	…
n=20:	[0.666667, 0.933332, 1, 0]
n=21:	[0.666667, 0.933332, 1, 0]
	$\approx$ [2/3, 14/15, 1, 0]

Example – Value iteration + LP



	$[\ x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}\ ]$
n=0:	[0.000000, 0.000000, 1, 0]
n=1:	[0.000000, 0.400000, 1, 0]
n=2:	[0.400000, 0.600000, 1, 0]
n=3:	[0.600000, 0.740000, 1, 0]
n=4:	[0.650000, 0.830000, 1, 0]
n=5:	[0.662500, 0.880000, 1, 0]
n=6:	[0.665625, 0.906250, 1, 0]
n=7:	[0.666406, 0.919688, 1, 0]
n=8:	[0.666602, 0.926484, 1, 0]
n=9:	[0.666650, 0.929902, 1, 0]
	…
n=20:	[0.666667, 0.933332, 1, 0]
n=21:	[0.666667, 0.933332, 1, 0]
	≈ [2/3, 14/15, 1, 0]

Method 3 – Policy iteration

- Value iteration:
  - iterates over (vectors of) probabilities
- Policy iteration:
  - iterates over adversaries ("policies")

- 1. Start with an arbitrary (memoryless) adversary A
- 2. Compute the reachability probabilities $\underline{\text{Prob}}^{A}(\text{F a})$ for A
- 3. Improve the adversary in each state
- 4. Repeat 2/3 until no change in adversary

- Termination:
  - finite number of memoryless adversaries
  - improvement in (minimum) probabilities each time

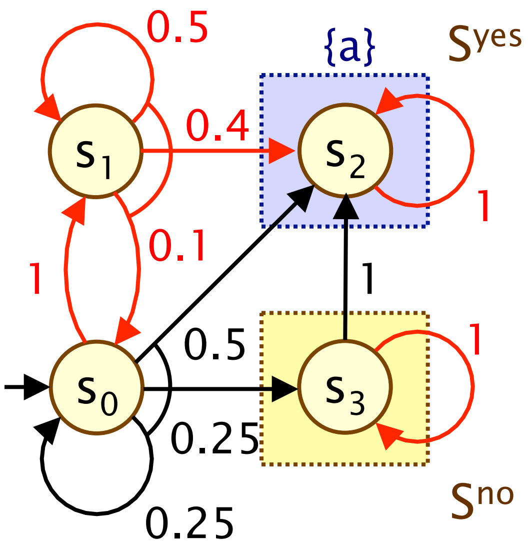
Method 3 – Policy iteration

- 1. Start with an arbitrary (memoryless) adversary A
  - pick some **Steps**(s) for each state $s \in S$
- 2. Compute the reachability probabilities Prob^A(F a) for A
  - probabilistic reachability on a DTMC
  - i.e. solve linear equation system
- 3. Improve the adversary in each state

$$A'(s) = \mathrm{argmin} \left\{ \sum_{s' \in S} \mu(s') \cdot \mathrm{Prob}^A(s', F\ a) \mid (a, \mu) \in \mathbf{Steps}(s) \right\}$$

- 4. Repeat 2/3 until no change in adversary

Example – Policy iteration



Arbitrary policy A:

Compute: $\mathrm{Prob}^A(F\ a)$

Let $x_i = \mathrm{Prob}^A(s_i, F\ a)$

$x_2=1$, $x_3=0$ and:

- $x_0 = x_1$
- $x_1 = 0.1 \cdot x_0 + 0.5 \cdot x_1 + 0.4$

Solution:

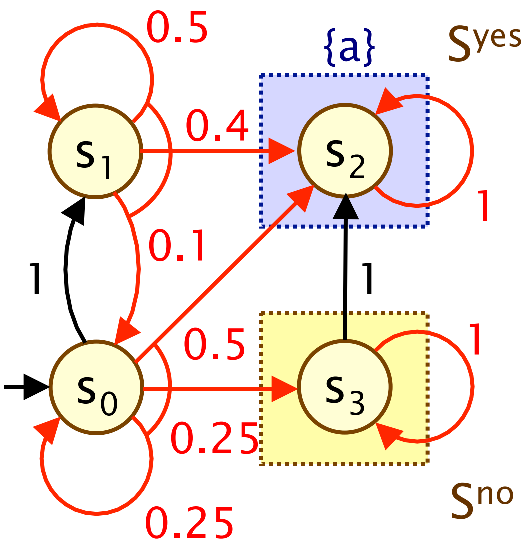
$\mathrm{Prob}^A(F\ a) = [\ 1,\ 1,\ 1,\ 0\ ]$

Refine A in state s_0:

$\min\{1(1),\ 0.5(1)+0.25(0)+0.25(1)\}$

$= \min\{1,\ 0.75\} = 0.75$

Example – Policy iteration



Refined policy A':

Compute: $\underline{Prob}^{A'}(F\ a)$

Let $x_i = Prob^{A'}(s_i, F\ a)$

$x_2=1$, $x_3=0$ and:

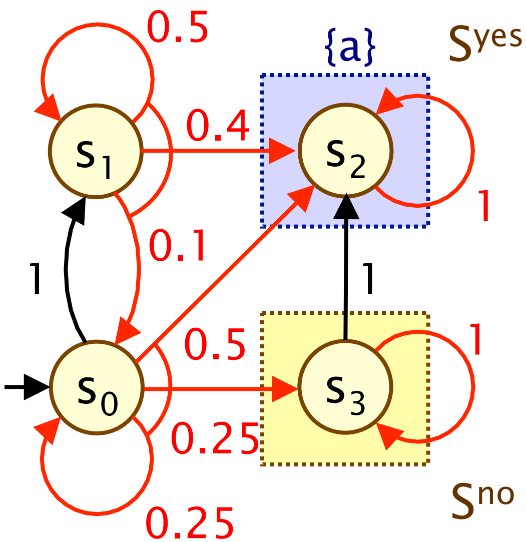
- $x_0 = 0.25 \cdot x_0 + 0.5$
- $x_1 = 0.1 \cdot x_0 + 0.5 \cdot x_1 + 0.4$

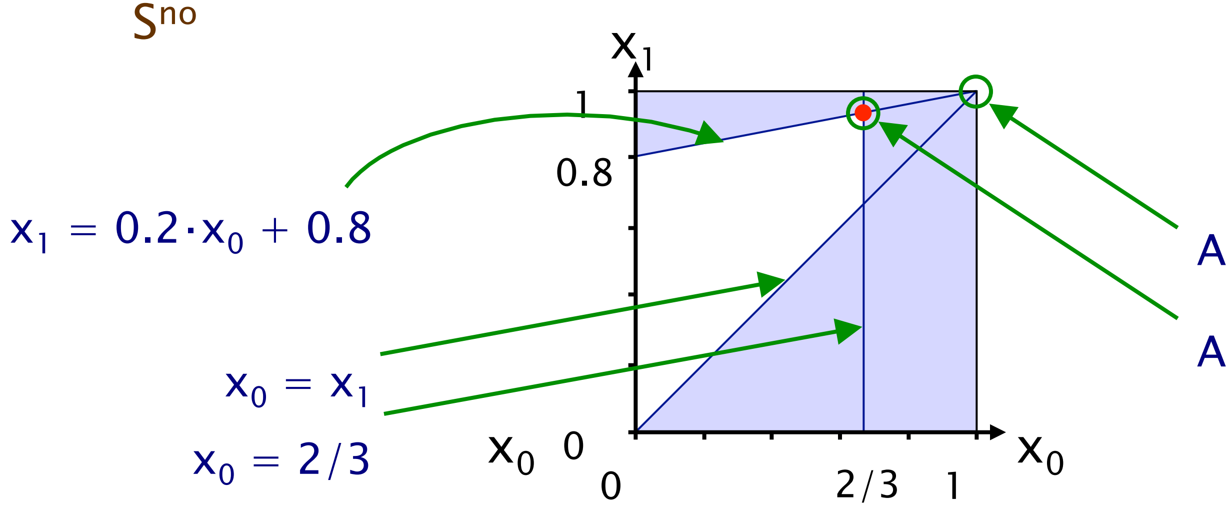
Solution:

$\underline{Prob}^{A'}(F\ a) = [\ 2/3,\ 14/15,\ 1,\ 0\ ]$

This is optimal

Example – Policy iteration





PCTL model checking – Summary

- Computation of set Sat(Φ) for MDP M and PCTL formula Φ
  - recursive descent of parse tree
  - combination of graph algorithms, numerical computation

- Probabilistic operator P:
  - X Φ : one matrix-vector multiplication, $O(|S|^2)$
  - Φ_1 $U^{\leq k}$ Φ_2 : k matrix-vector multiplications, $O(k|S|^2)$
  - Φ_1 U Φ_2 : linear programming problem, polynomial in |S| (assuming use of linear programming)

- Complexity:
  - linear in |Φ| and polynomial in |S|
  - S is states in MDP, assume |Steps(s)| is constant

Overview (Part 2)

- Markov decision processes (MDPs)
- Adversaries & probability spaces
- PCTL for MDPs
- PCTL model checking
- Further model checking (LTL, costs & rewards)
- Case study: Firewire root contention

LTL model checking for MDPs

- We consider lower/upper bounds for an LTL formula ψ:
  - $p_{min}(s, \psi) = \inf_{A \in Adv} Prob^A(s, \psi)$
  - $p_{max}(s, \psi) = \sup_{A \in Adv} Prob^A(s, \psi)$

- To model check an LTL formula ψ on an MDP M
  - 1. Convert problem to one needing maximum probabilities
    - $p_{min}(s, \psi) = 1 - p_{max}(s, \neg\psi)$
  - 2. Construct product MDP $M \otimes A$
    - of MDP M and deterministic Rabin automaton A for ψ (or ¬ψ)
  - 3. Identify accepting end components (ECs) of $M \otimes A$
    - an end component is the analogue of a bottom strongly connected component (BSCC) in a DTMC
  - 4. Compute maximum probability of reaching accepting ECs

- Complexity: doubly exponential in |ψ|, polynomial in |M|

Costs and rewards for MDPs

- Can use costs and rewards in similar fashion to DTMCs:

- Augment MDPs with rewards (or costs)
  - (but often assign to states/actions, not states/transitions)

- Extend logic PCTL with R operator
  - semantics extended in same way as P operator
  - e.g. $s \vDash R_{\sim r}\ [\ F\ \Phi\ ] \ \Leftrightarrow\ Exp^{A}(s, X_{F\Phi}) \sim r$ for all adversaries A
  - quantitative properties: $Rmin_{=?}$ [...] and $Rmax_{=?}$ [...]

- Examples:
  - "the minimum expected queue size after exactly 90 seconds"
  - "the maximum expected power consumption over one hour"
  - the maximum expected time for the algorithm to terminate

Model checking MDP reward formulas

- Instantaneous: $R_{\sim r}$ [$I^{=k}$]
  - similar to the computation of bounded until probabilities
  - solution of recursive equations
- Cumulative: $R_{\sim r}$ [$C^{\leq k}$]
  - extension of bounded until computation
  - solution of recursive equations
- Reachability: $R_{\sim r}$ [F ϕ]
  - similar to the case for P operator and until
  - graph-based precomputation (identify ∞-reward states)
  - then linear programming problem (or value iteration)

Overview (Part 2)

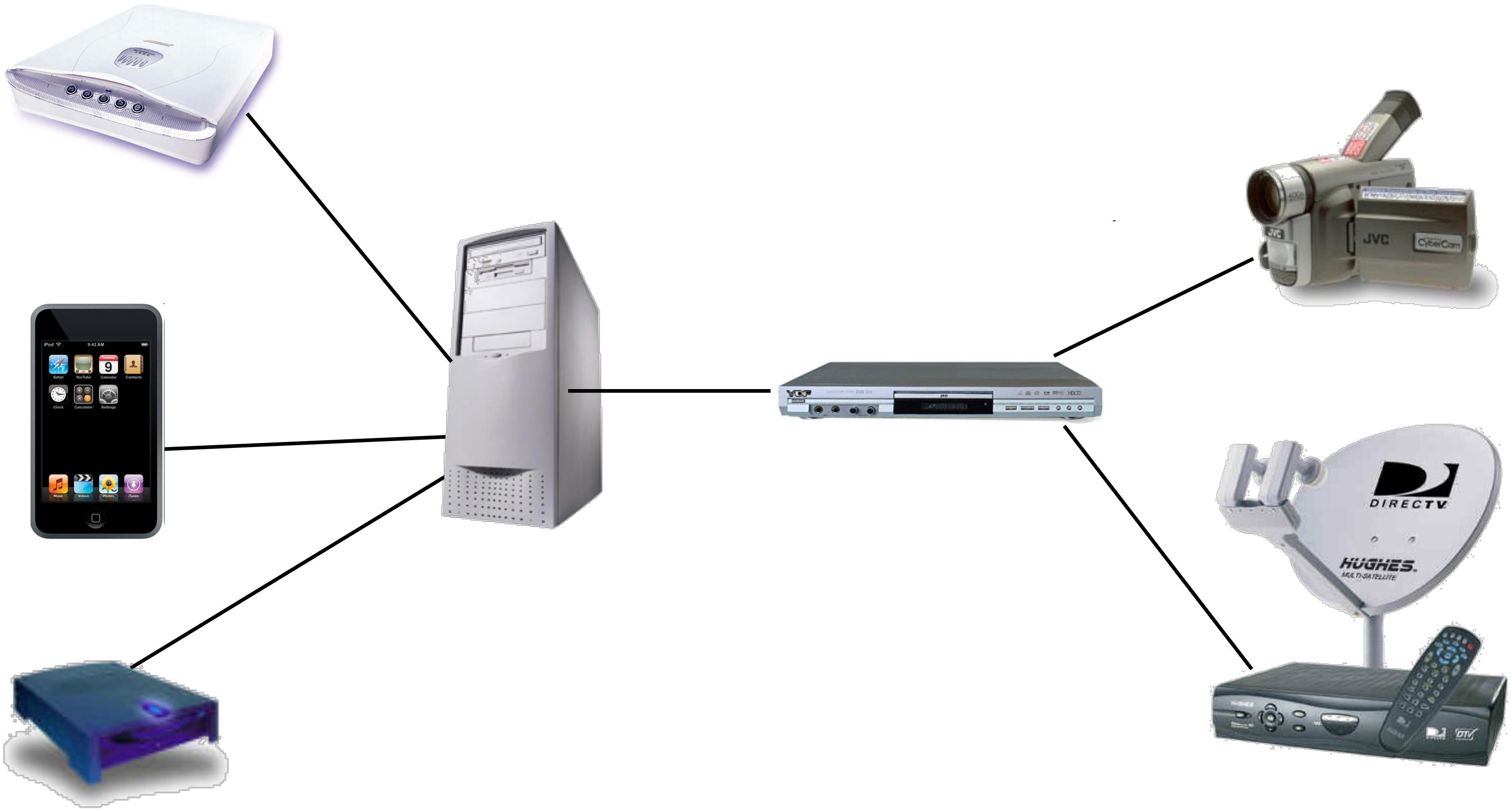
- Markov decision processes (MDPs)
- Adversaries & probability spaces
- PCTL for MDPs
- PCTL model checking
- Further model checking (LTL, costs & rewards)
- Case study: Firewire root contention

Case study: FireWire protocol

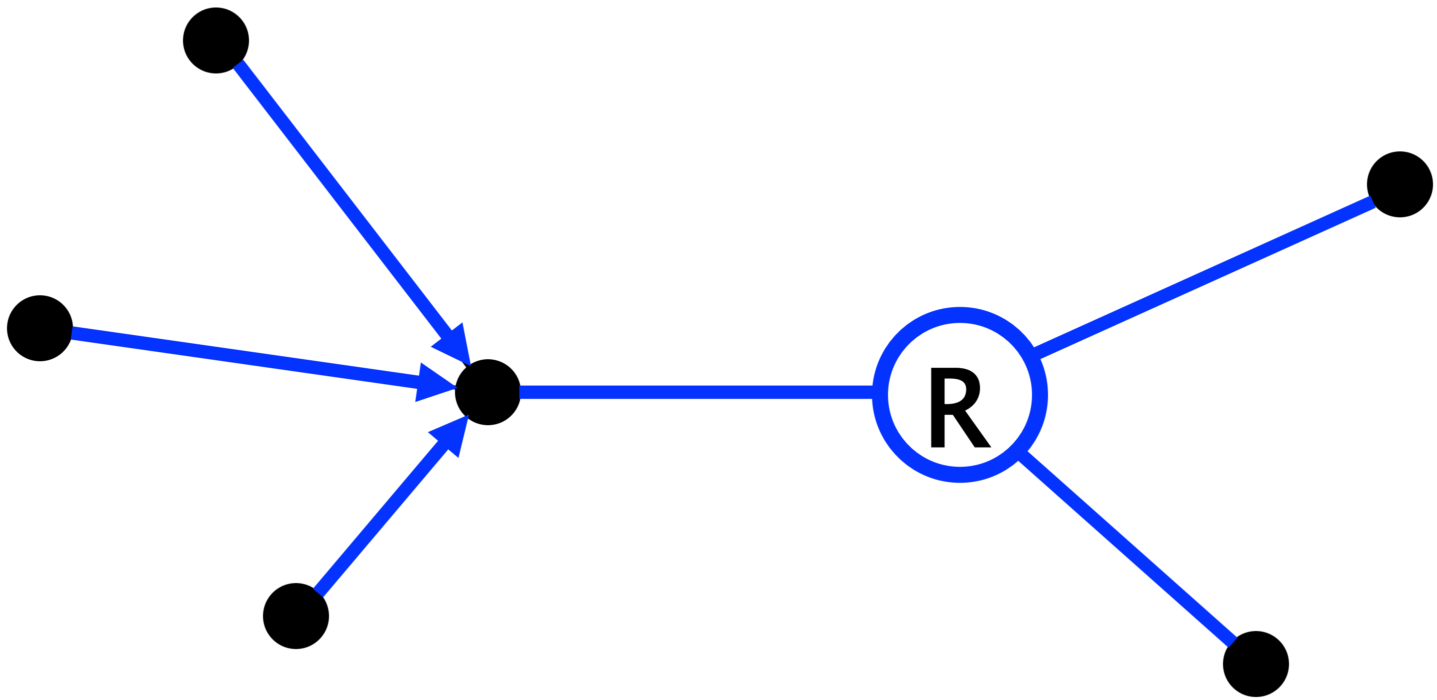
- FireWire (IEEE 1394)
  - high-performance serial bus for networking multimedia devices; originally by Apple
  - "hot-pluggable" - add/remove devices at any time
  - no requirement for a single PC (need acyclic topology)

- Root contention protocol
  - leader election algorithm, when nodes join/leave
  - symmetric, distributed protocol
  - uses electronic coin tossing and timing delays
  - nodes send messages: "be my parent"
  - root contention: when nodes contend leadership
  - random choice: "fast"/"slow" delay before retry

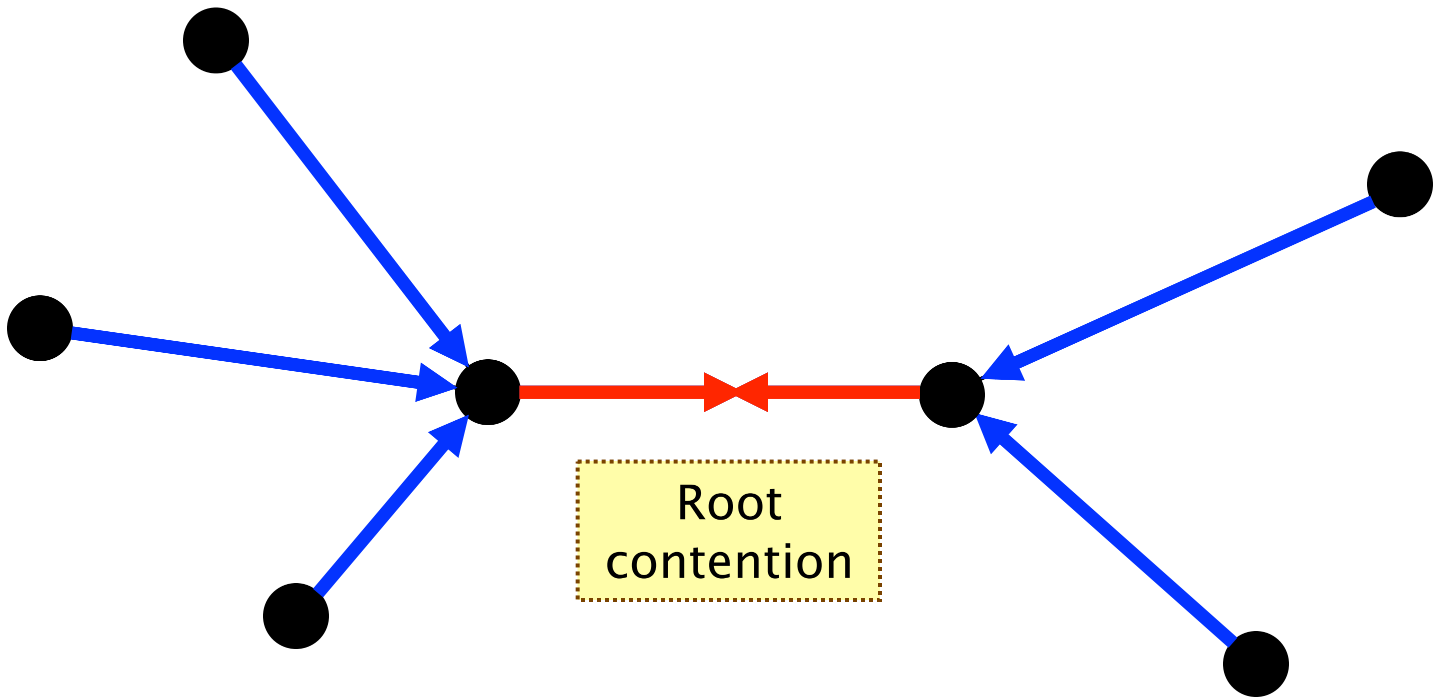
FireWire example



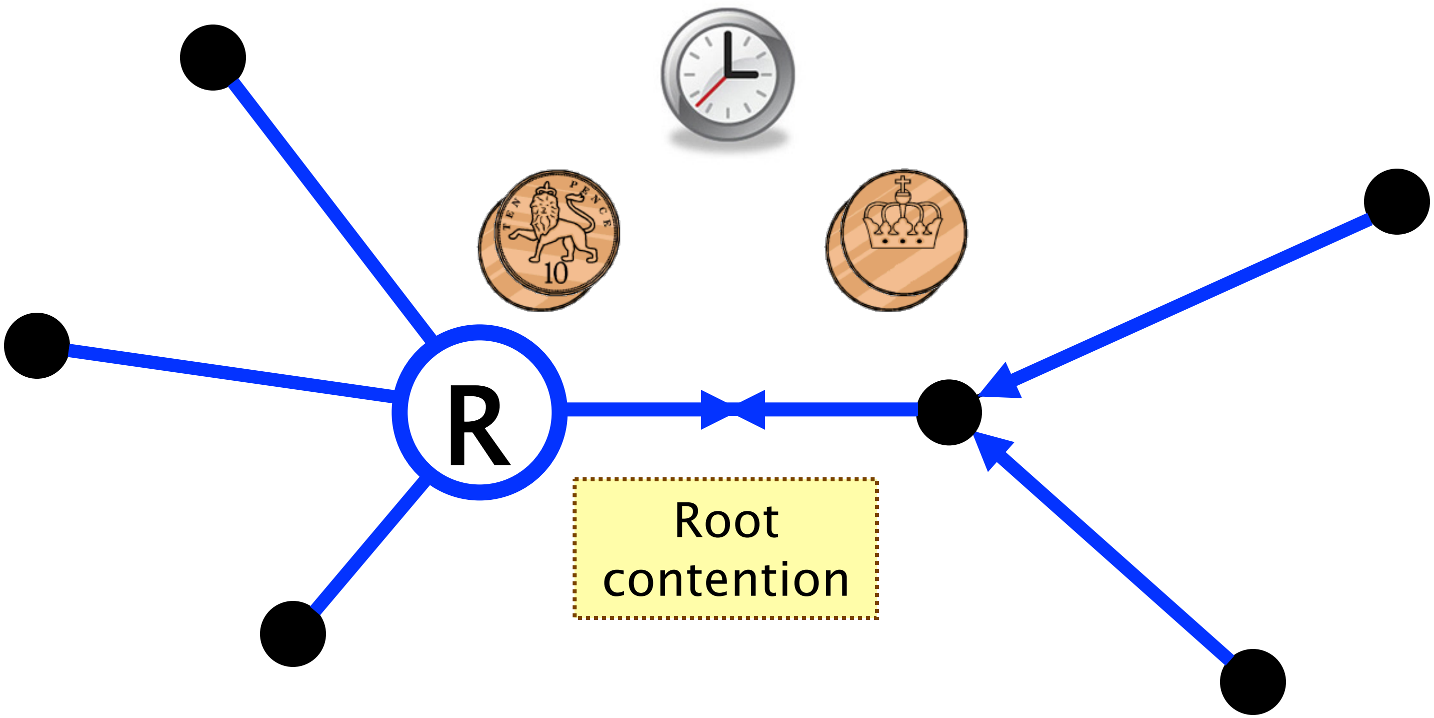
FireWire leader election



FireWire root contention

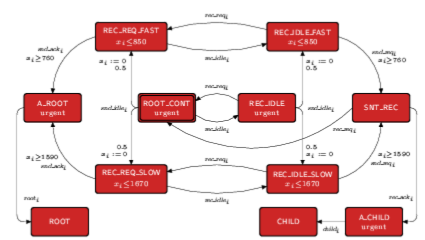


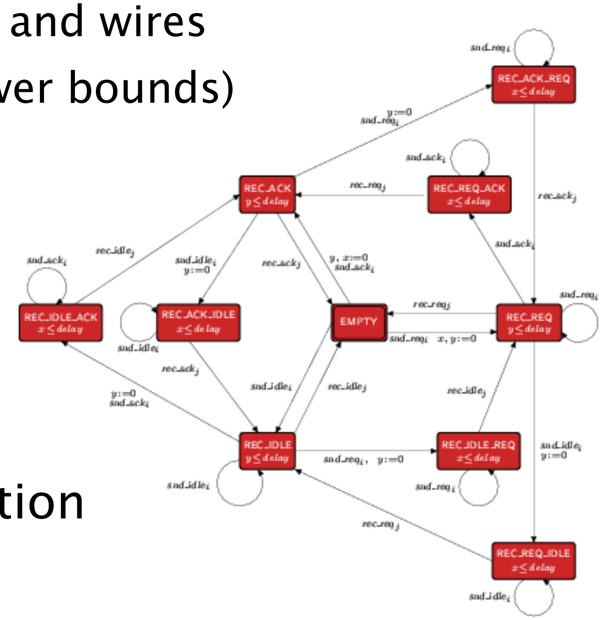
FireWire root contention



FireWire analysis

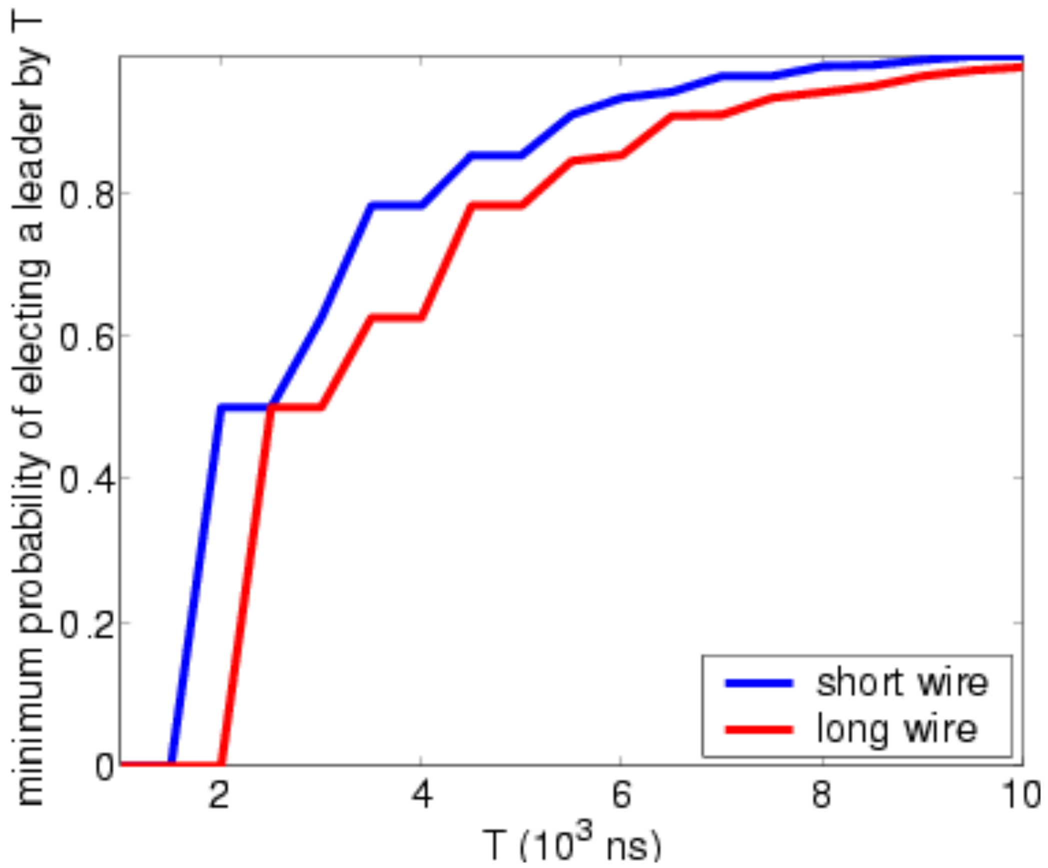
- Probabilistic model checking
  - model constructed and analysed using PRISM
  - timing delays taken from standard
  - model includes:
    - concurrency: messages between nodes and wires
    - underspecification of delays (upper/lower bounds)
  - max. model size: 170 million states





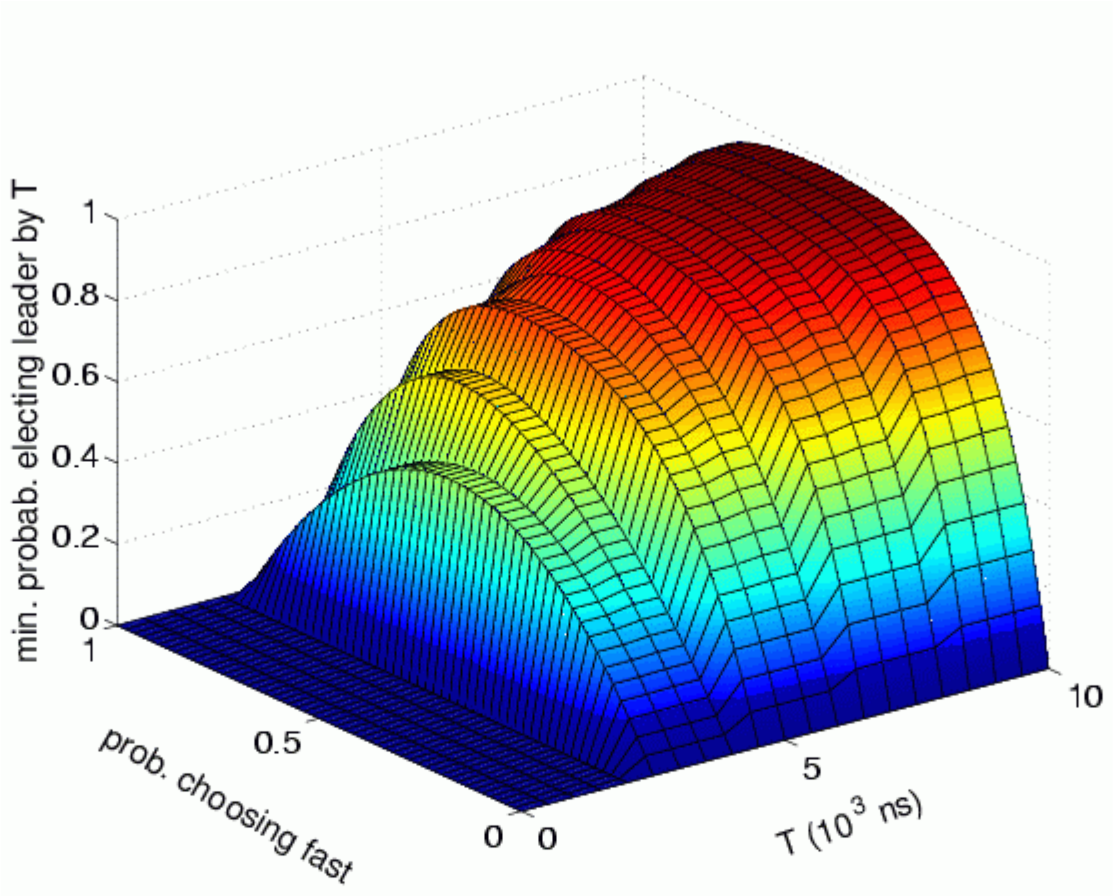
- Analysis:
  - verified that root contention always resolved with probability 1
  - investigated time taken for leader election
  - and the effect of using biased coin
    - based on a conjecture by Stoelinga

FireWire: Analysis results



"minimum probability of electing leader by time T"

FireWire: Analysis results

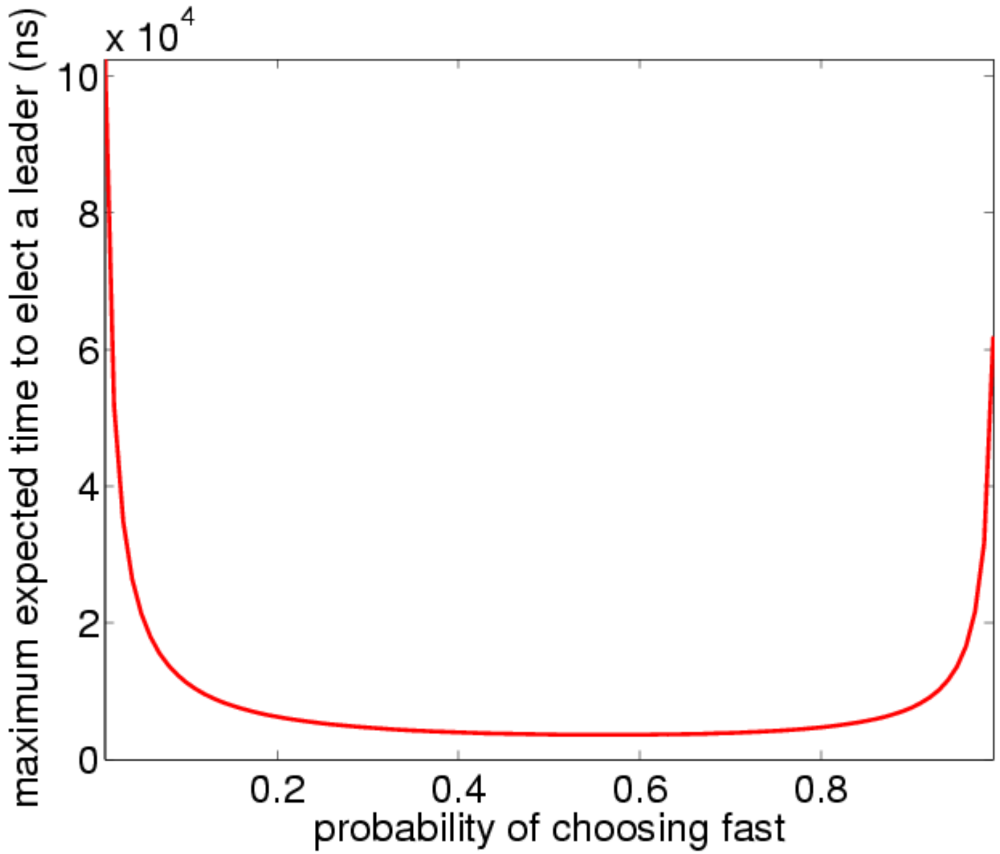


"minimum probability of electing leader by time T"

(short wire length)

Using a biased coin

FireWire: Analysis results

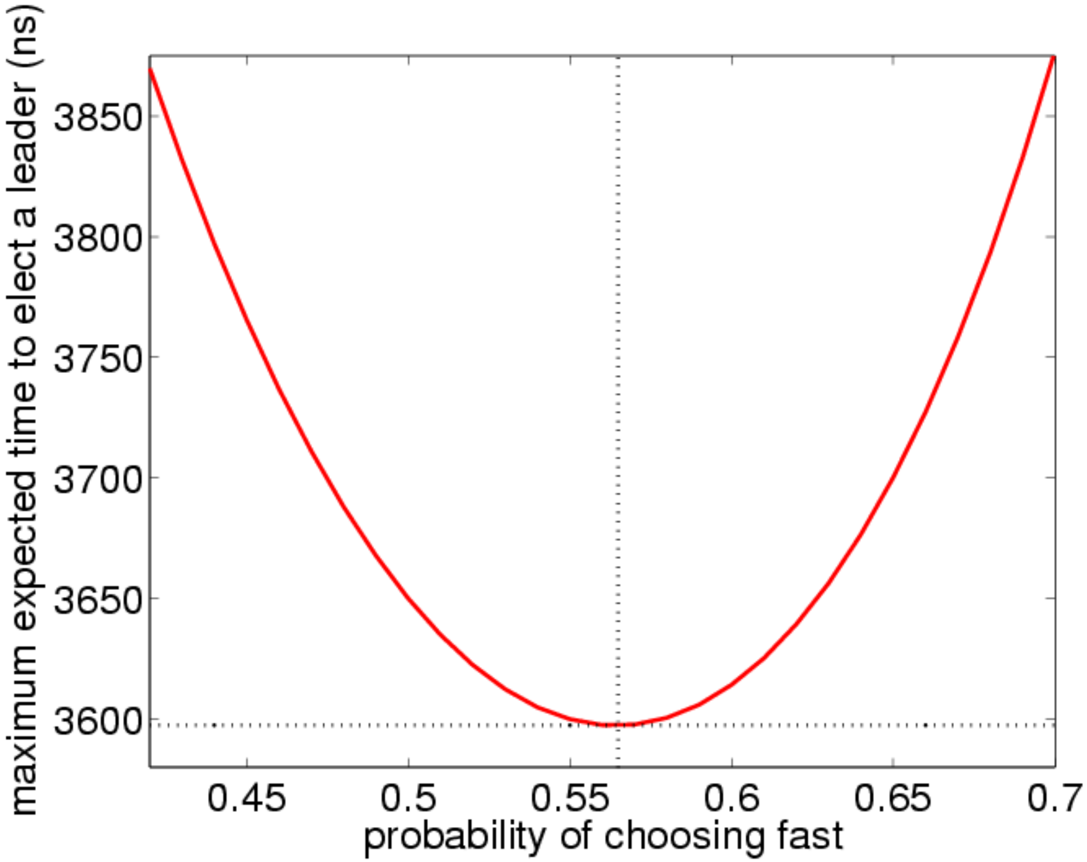


"maximum expected time to elect a leader"

(short wire length)

Using a biased coin

FireWire: Analysis results



"maximum expected time to elect a leader"

(short wire length)

Using a biased coin is beneficial!

Summary

- Markov decision processes (MDPs)
  - extend DTMCs with nondeterminism
  - to model concurrency, underspecification, …
- An adversary resolve nondeterminism in an MDP
  - induce a probability space over paths
  - consider minimum/maximum probabilities over all adversaries
- Property specifications
  - use e.g. PCTL or LTL, as for DTMCs
  - but quantify over all adversaries
- Model checking algorithms
  - covered three basic techniques for MDPs: linear programming, value iteration, or policy iteration

- Tomorrow: continuous-time Markov chains (CTMCs)