Lecture 16
Strategy synthesis for MDPs

Alessandro Abate

Department of Computer Science
University of Oxford

Overview

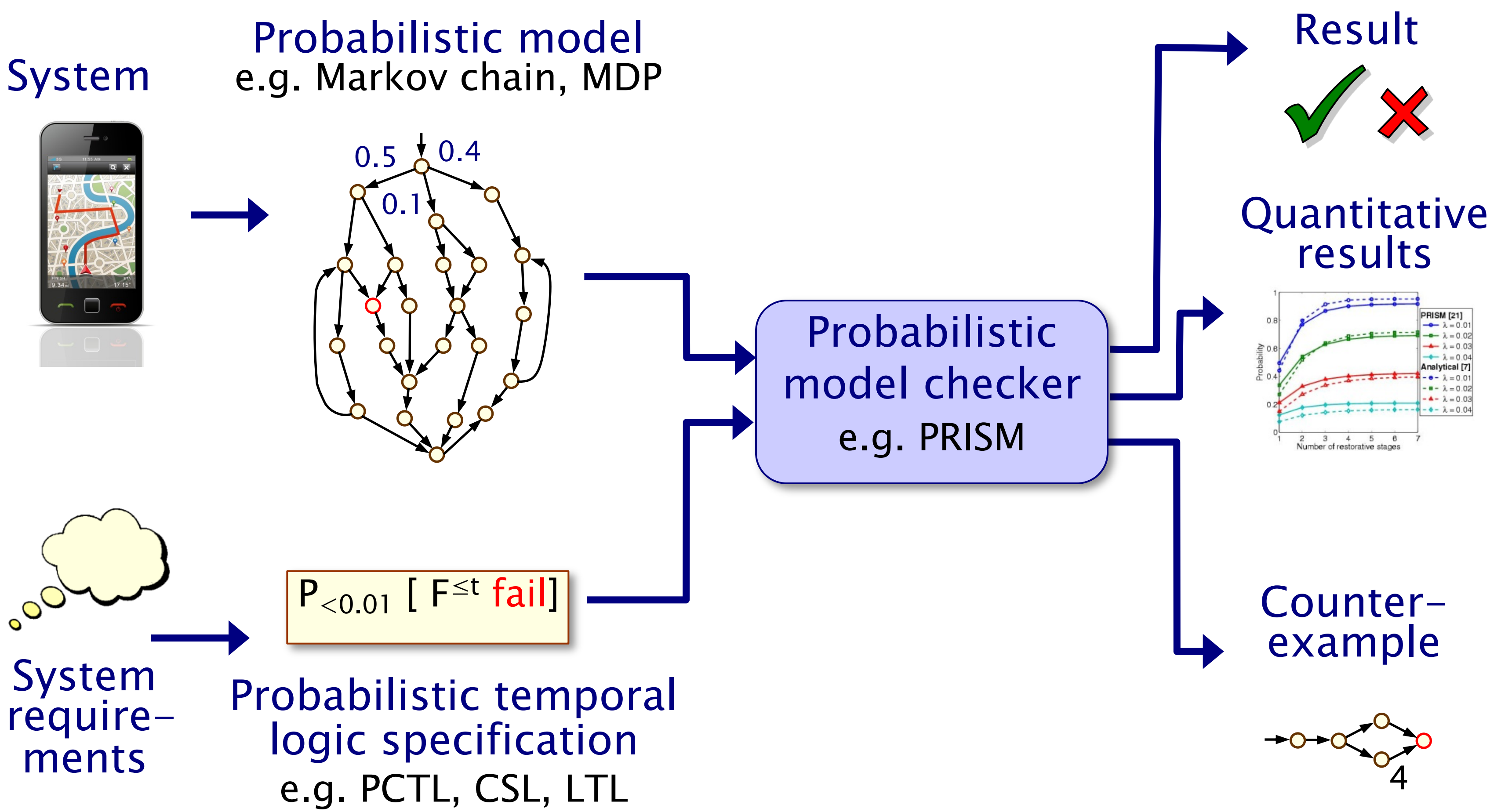
- Recall: MDPs, adversaries, properties and objectives
- The strategy synthesis problem
  - (also known as adversary, scheduler, policy, controller)
- Strategy synthesis
  - for probabilistic reachability
  - for probabilistic/reward LTL properties
  - for multi-objective LTL properties

About this lecture…

- So far have focused on verification
  - probabilistic models
  - quantitative temporal specifications
  - model checking algorithms
- Some work to date on counterexamples
  - but difficult to represent them compactly
- We consider the problem of strategy synthesis for MDPs
  - can we find a strategy to guarantee that a given quantitative property is satisfied?
  - advantage: correct-by-construction procedure
  - incidentally, can reuse verification algorithms…
- More generally, shifting emphasis
  - from quantitative verification to quantitative synthesis

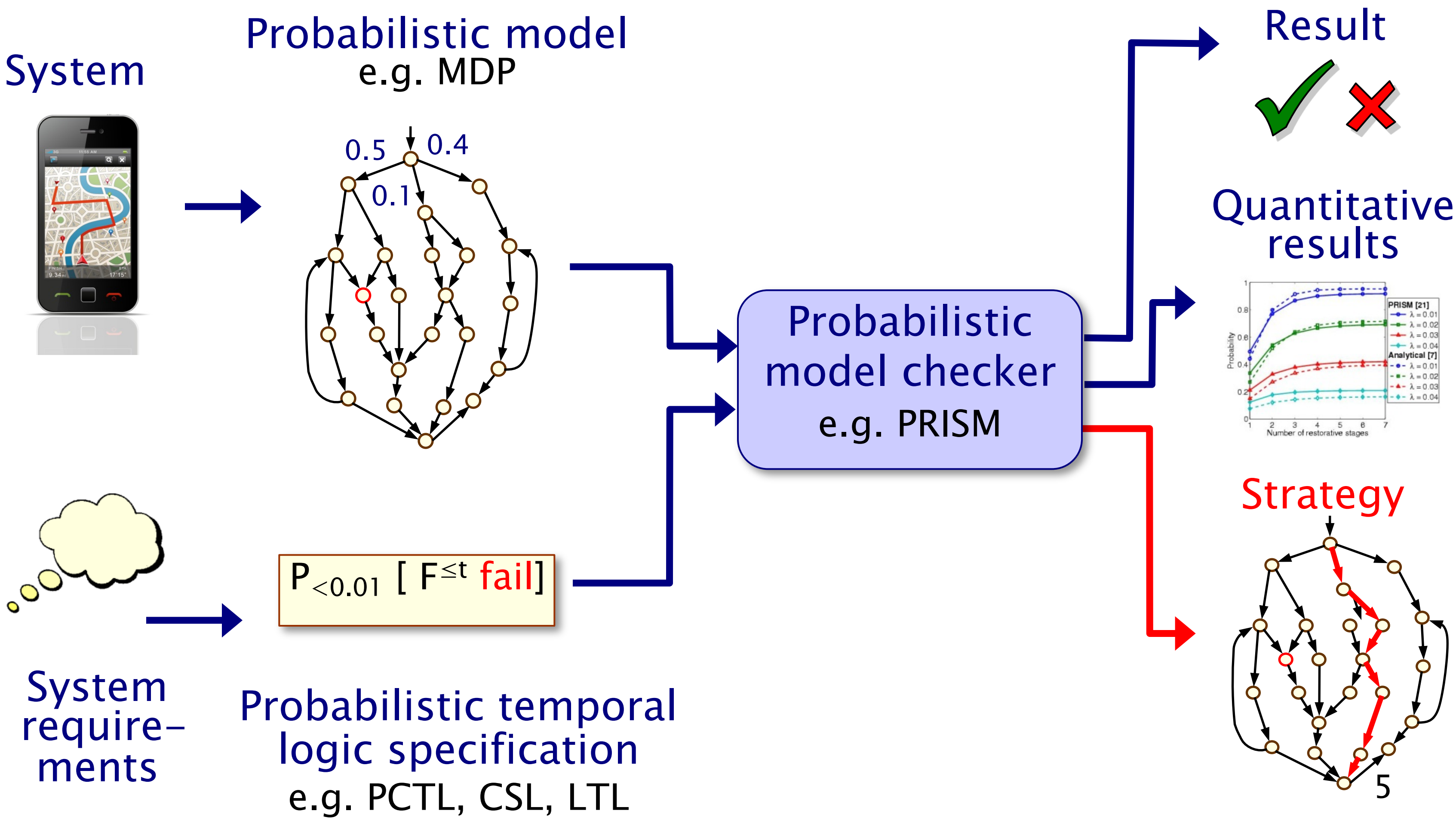
Quantitative (probabilistic) verification

Automatic verification (model checking) of quantitative properties of probabilistic models of system



Quantitative (probabilistic) verification

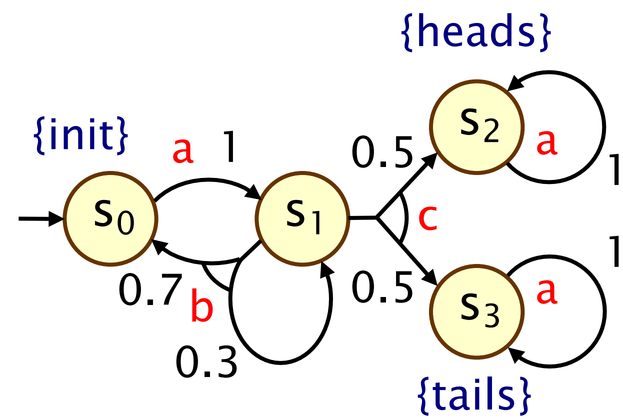
Automatic verification and strategy synthesis over quantitative properties for probabilistic models



Recall: Markov decision processes (MDPs)

- Model nondeterministic as well as probabilistic behaviour
  - e.g. for concurrency, under-specification, abstraction…
  - extension of discrete-time Markov chains
  - nondeterministic choice between probability distributions

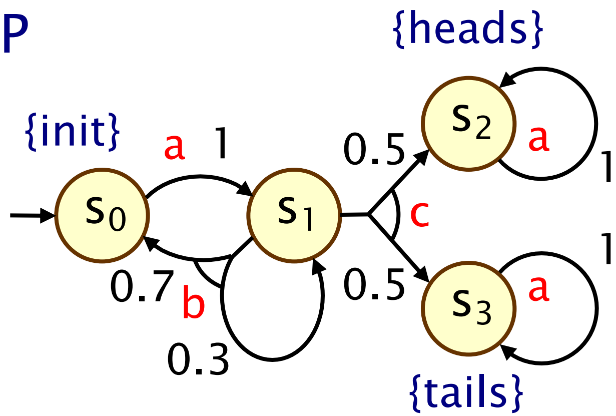
- Formally, an MDP is a tuple
  - $(S, s_{init}, \mathbf{Steps}, L)$



- where:
  - S is a set of states
  - $s_{init} \in S$ is the initial state
  - $\mathbf{Steps} : S \to 2^{Act \times Dist(S)}$ is a transition probability function
  - $L : S \to 2^{AP}$ is a labelling function
  - Act is a set of actions, Dist(S) is the set of probability distributions over S, AP is a set of atomic propositions

Paths and strategies

- A (finite or infinite) path through an MDP
  - is a sequence (s_0...s_n) of (connected) states
  - represents an execution of the system
  - resolves both probabilistic and nondeterministic choices



- A strategy σ (aka. "adversary" or "policy") of an MDP
  - is a resolution of nondeterminism only
  - is (formally) a mapping from finite paths to distributions over steps enabled in the last state of the path
  - induces a fully probabilistic model
  - i.e. an (infinite-state) Markov chain over finite paths
  - on which we can define a probability space over infinite paths

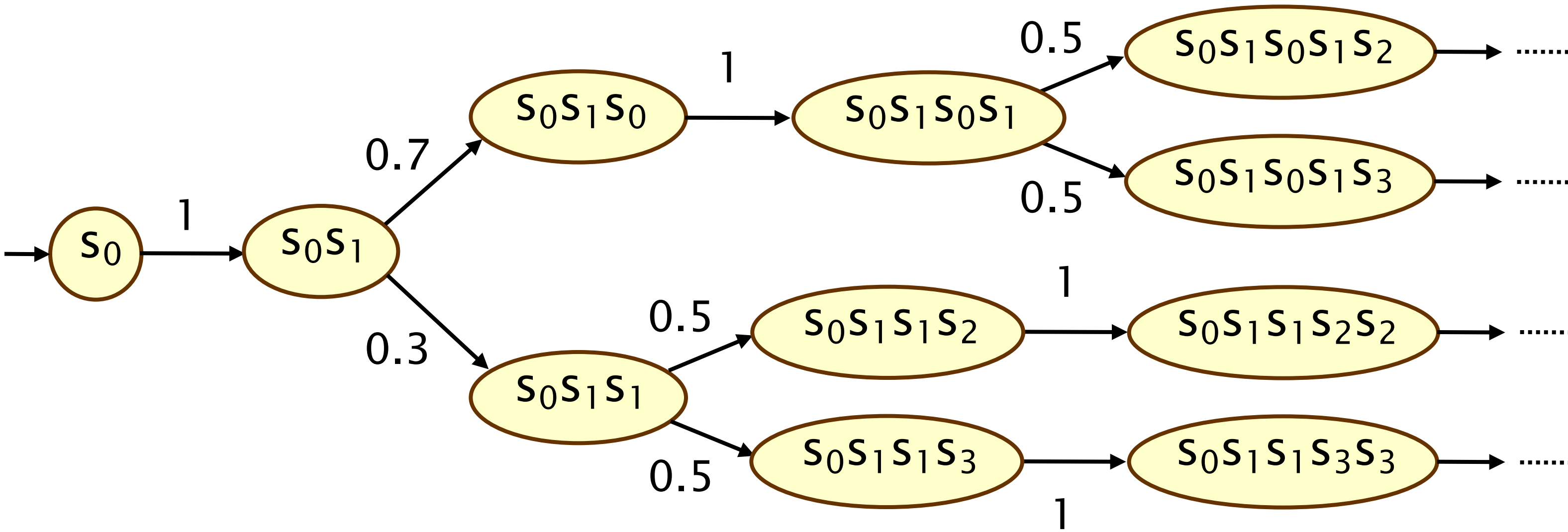
Classification of strategies/adversaries

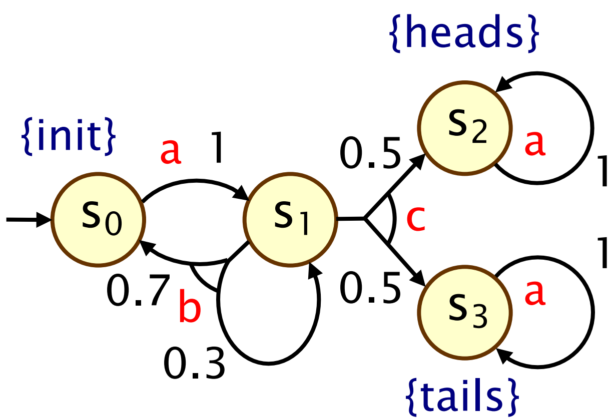
- Strategies are classified according to
  - randomisation:
    - σ is deterministic (pure) if $\sigma(s_0 \ldots s_n)$ is a point distribution, and randomised otherwise
  - memory:
    - σ is memoryless (simple) if $\sigma(s_0 \ldots s_n) = \sigma(s_n)$ for all $s_0 \ldots s_n$
    - σ is finite memory if there are finitely many *modes* such that $\sigma(s_0 \ldots s_n)$ depends only on s_n and *current mode*, which is updated each time an action is performed
    - otherwise, σ is infinite memory

- A strategy σ induces, for each state s in the MDP:
  - a set of infinite paths $Path^{\sigma}(s)$
  - a probability space $Prob^{\sigma}_{s}$ over $Path^{\sigma}(s)$

Example strategy

- Fragment of induced Markov chain for strategy which picks b then c in s_1





finite-memory,
deterministic

Costs and rewards

- We can augment MDPs with rewards (or costs)
  - real-valued quantities assigned to states and/or actions
  - different from the DTMC case where transition rewards assigned to individual transitions
- MDP $(S, s_{init}, \mathbf{Steps}, L)$, a reward/cost structure is a pair $(\underline{\rho}, \iota)$
  - $\underline{\rho} : S \rightarrow \mathbb{R}_{\geq 0}$ is the state reward function
  - $\iota : S \times Act \rightarrow \mathbb{R}_{\geq 0}$ is transition reward function
- These can be used to compute:
  - elapsed time, power consumption, size of message queue, number of messages successfully delivered, net profit, …
- Distinguish between types of rewards over paths
  - Instantaneous ($I^{=k}$)
  - Cumulative and Bounded-cumulative (C, $C^{\leq k}$)
  - Reachability ($F\ \phi$)

Properties and objectives

- The syntax:

ψ is true with probability ~p

expected reward is ~r

  - ϕ ::= $P_{\sim p}$ [ψ] | $R_{\sim r}$ [ρ]
  - ψ ::= true | a | ψ ∧ ψ | ¬ ψ | X ψ | ψ $U^{\leq k}$ ψ | ψ U ψ

"next" "bounded until" "until"

  - ρ ::= F b | C | $C^{\leq k}$ | $I^{=k}$

"reachability" "cumulative" "instantaneous"

  - where b is an atomic proposition, used to identify states of interest, $p \in [0,1]$ is a probability, $\sim \in \{<,>,\leq,\geq\}$, $k \in \mathbb{N}$, and $r \in \mathbb{R}_{\geq 0}$
- We refer to ϕ as property, ψ and ρ as objectives
  - (linear-time: branching time more challenging for synthesis)

Properties and objectives

- Semantics of the probabilistic operator P
  - $s \models P_{\sim p}$ [ψ] means "the probability, from state s, that ψ is true for an outgoing path satisfies ~p for all strategies σ"
  - formally $s \models P_{\sim p} [\ \psi\ ] \Leftrightarrow Pr_s^\sigma(\psi) \sim p$ for all strategies σ
  - where we use $Pr_s^\sigma(\psi)$ to denote $Pr_s^\sigma \{ \omega \in Path_s^\sigma \mid \omega \models \psi \}$

- $R_{\sim r}$ [ρ] means "the expected value of ρ satisfies ~r for all strategies"

- Some examples:
  - $P_{\geq 0.4}$ [F "goal"] "probability of reaching goal is at least 0.4"
  - $R_{<5}$ [$C^{\leq 60}$] "expected power consumption over one hour is below 5"
  - $R_{\leq 10}$ [F "end"] "expected time to termination is at most 10"

Verification and strategy synthesis

- The verification problem is:
  - Given an MDP M and a property ϕ, does M satisfy ϕ under any possible strategy σ?
- The synthesis problem is dual:
  - Given an MDP M and a property ϕ, find, if it exists, a strategy σ such that M satisfies ϕ under σ
- Verification and strategy synthesis is achieved using the same techniques, namely computing optimal values for probability objectives:
  - $p_{min}(s,\psi) = \inf_\sigma Prob^\sigma(s,\psi)$
  - $p_{max}(s,\psi) = \sup_\sigma Prob^\sigma(s,\psi)$
  - and similarly for expectations

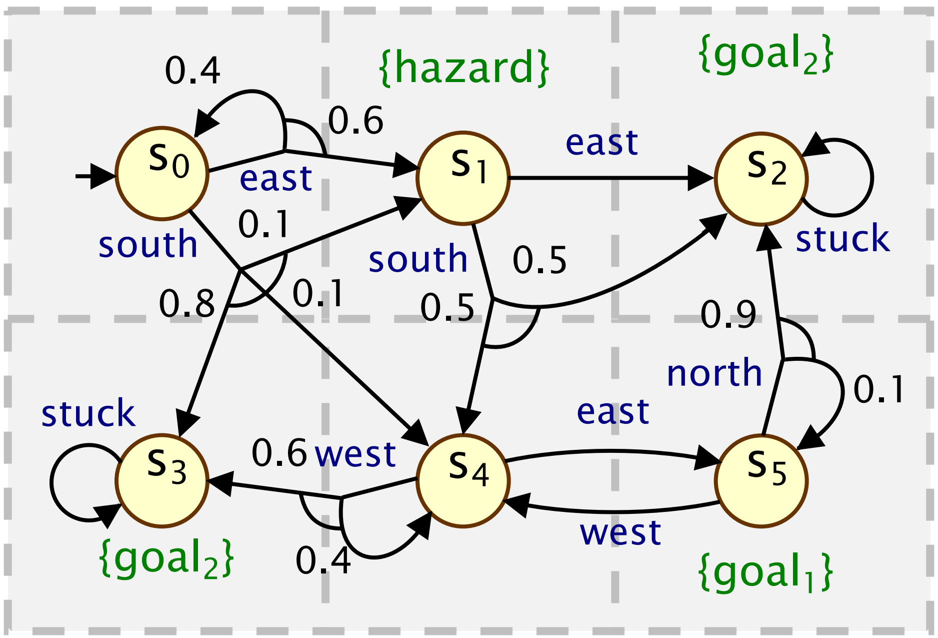
Computing prob. reachability for MDPs

- Computation of $p_{max}(s,F\ b)$ for all $s \in S$ (for p_{min} analogous)
- Step 1: pre-compute all states where probability is 1 or 0
  - graph-based algorithm, yielding sets S^{yes}, S^{no}
- Step 2: compute probabilities for remaining states ($S^?$)
  - (i) approximate with value iteration
  - (ii) solve linear programming problem
  - (iii) solve with policy (strategy) iteration

- 1. Precomputation, e.g.:
  - algorithm Prob1E computes S^{yes}
    - there exists a strategy for which the probability of "F b" is 1
  - algorithm Prob0A computes S^{no}
    - for all strategies, the probability of satisfying "F b" is 0

Running example

- Example MDP
  - robot moving through terrain divided into 3 x 2 grid



States:

s_0, s_1, s_2, s_3, s_4, s_5

Actions:

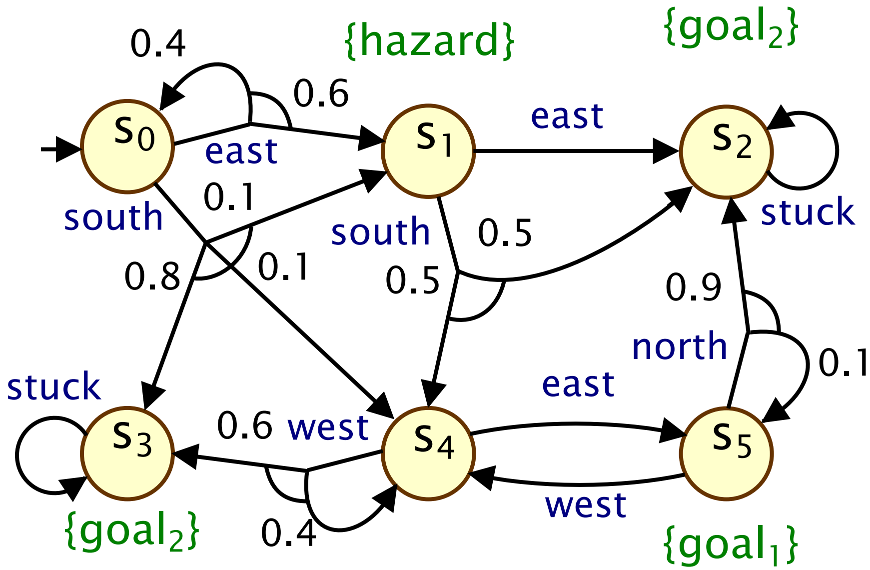
north, east, south, west, stuck

Labels

(atomic propositions):

hazard, goal$_1$, goal$_2$

Example - Reachability



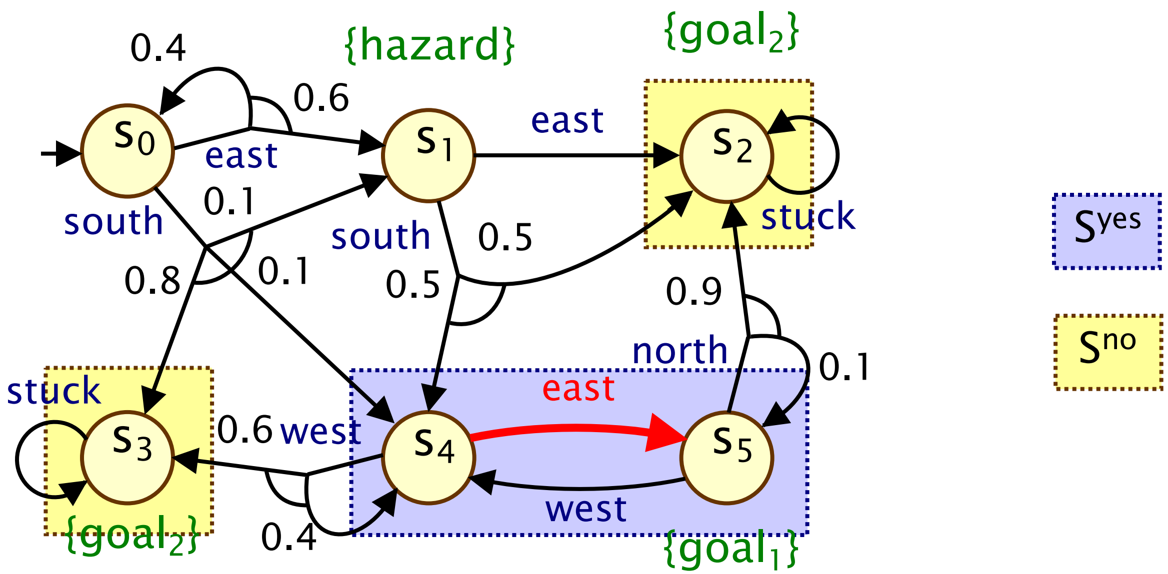
Example:

$P_{\geq 0.4}$ [F goal$_1$]

So compute:

p_{max}(s,F goal$_1$)

Example – Precomputation



Example:

$P_{\geq 0.4}$ [F goal$_1$]

So compute:

p_{max}(s,F goal$_1$)

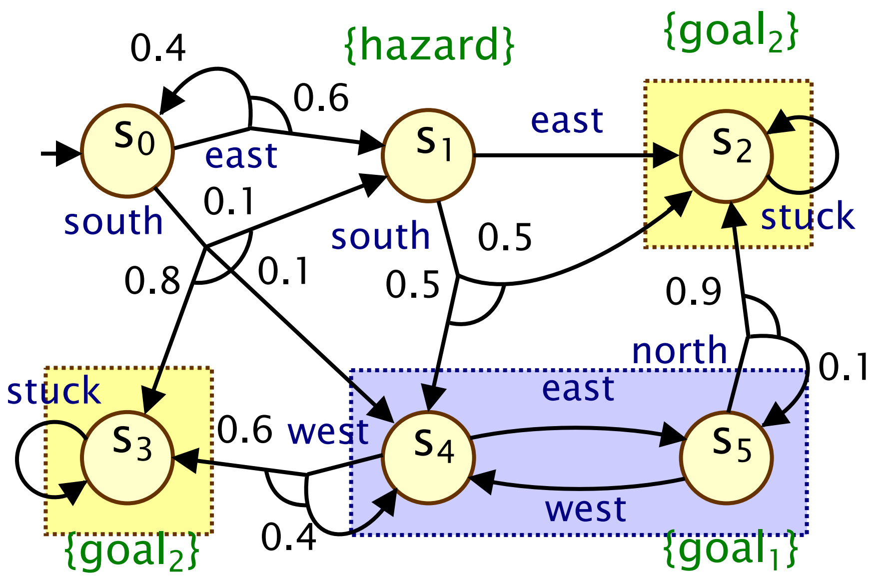
Reachability for MDPs

- 2. Numerical computation
  - compute probabilities $p_{max}(s, F\ b)$
  - for remaining states in $S^? = S \setminus (S^{yes} \cup S^{no})$
  - obtained as the unique solution of the linear programming (LP) problem:

$$\boxed{\begin{array}{c} \text{minimize } \sum_{s \in S^?} x_s \text{ subject to the constraints :} \\ x_s \geq \sum_{s' \in S^?} \mu(s') \cdot x_{s'} + \sum_{s' \in S^{yes}} \mu(s') \\ \text{for all } s \in S^? \text{ and for all } (a, \mu) \in \textbf{Steps}(s) \end{array}}$$

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

Example - Reachability (LP)



Let $x_i = p_{max}(s_i, F\ goal_1)$

S^{yes}: $x_4 = x_5 = 1$

S^{no}: $x_2 = x_3 = 0$

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

Minimise $x_0 + x_1$ subject to:

- $x_0 \geq 0.4 \cdot x_0 + 0.6 \cdot x_1$ (east)
- $x_0 \geq 0.1 \cdot x_1 + 0.1$ (south)
- $x_1 \geq 0.5$ (south)
- $x_1 \geq 0$ (east)

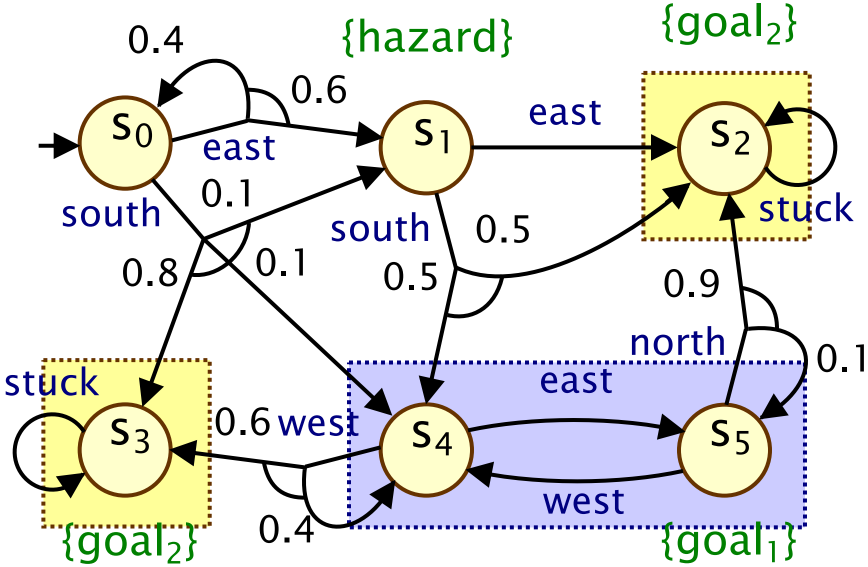
Example:

$P_{\geq 0.4}$ [F $goal_1$]

So compute:

$p_{max}(s, F\ goal_1)$

Example – Reachability (LP)



Let $x_i = p_{max}(s_i, F\ goal_1)$

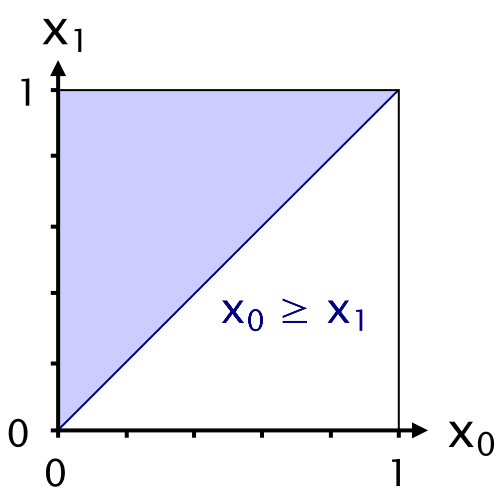
S^{yes}: $x_4 = x_5 = 1$

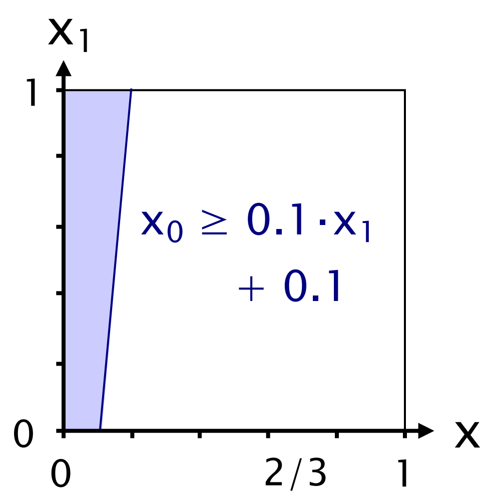
S^{no}: $x_2 = x_3 = 0$

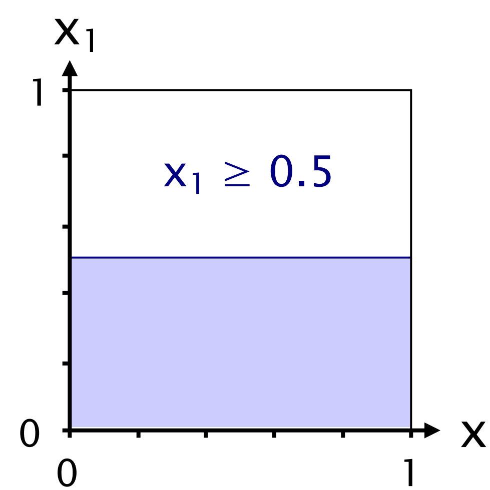
For $S^? = \{x_0, x_1\}$:

Minimise $x_0 + x_1$ subject to:

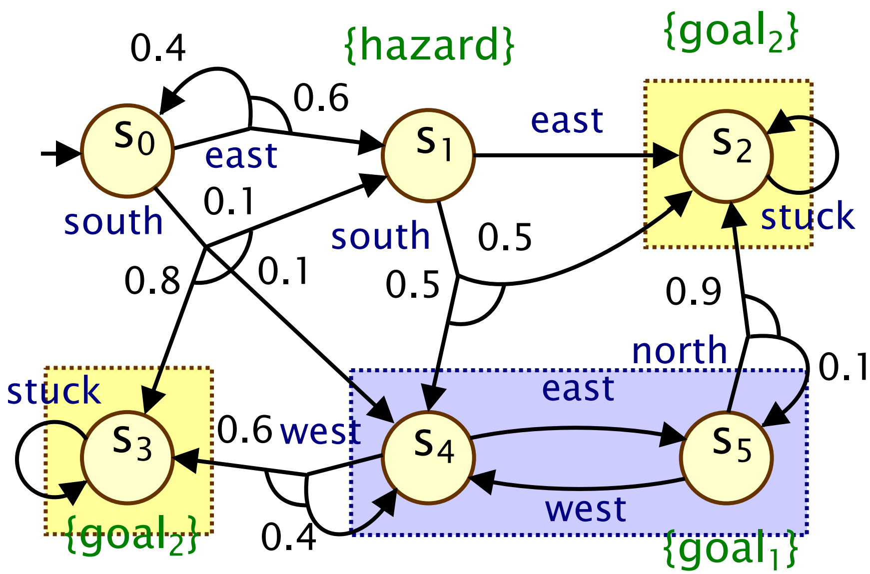
- $x_0 \geq x_1$ (east)
- $x_0 \geq 0.1 \cdot x_1 + 0.1$ (south)
- $x_1 \geq 0.5$ (south)







Example - Reachability (LP)



Let $x_i = p_{max}(s_i, F\ goal_1)$

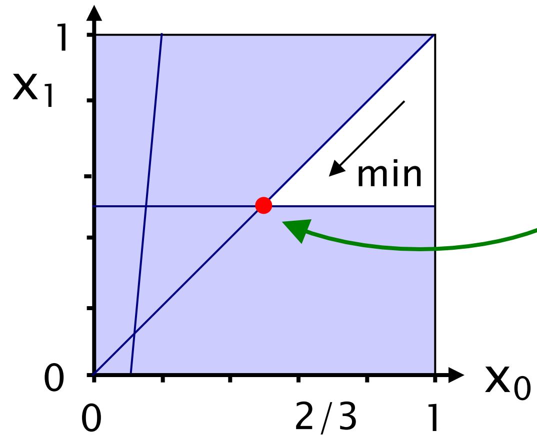
S^{yes}: $x_4 = x_5 = 1$

S^{no}: $x_2 = x_3 = 0$

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

Minimise $x_0 + x_1$ subject to:

- $x_0 \geq x_1$
- $x_0 \geq 0.1 \cdot x_1 + 0.1$
- $x_1 \geq 0.5$



Solution:

$(x_0, x_1) = (0.5, 0.5)$

i.e.

$p_{max}(s_0, F\ goal_1) = 0.5$

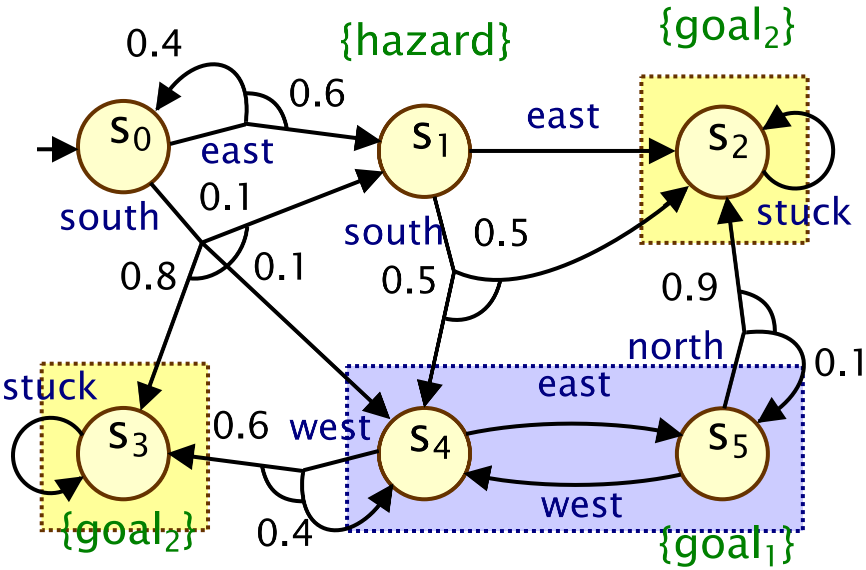
Reachability for MDPs

- 2. Numerical computation (alternative method)
  - value iteration
  - it can be shown that: $p_{max}(s, F\ b) = \lim_{n\to\infty} x_s^{(n)}$ where:

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

- Approximate iterative solution technique
  - iterations terminated when solution converges sufficiently

Example – Reachability (val. iter.)



Compute: $p_{max}(s, F\ goal_1)$

S^{yes}: $x_4 = x_5 = 1$

S^{no}: $x_2 = x_3 = 0$

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

$[\ x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}, x_4^{(n)}, x_5^{(n)}\ ]$

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

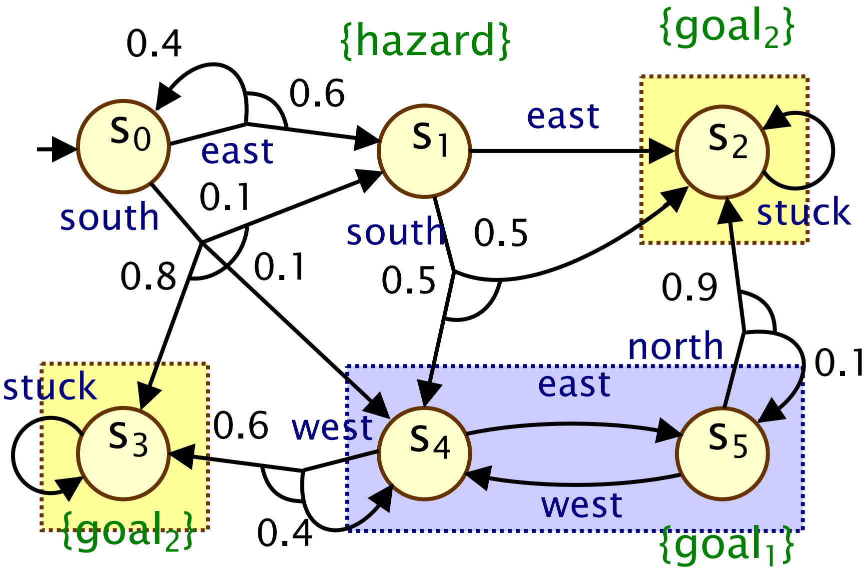
$n=1$: $[\ \max(0.6 \cdot 0 + 0.4 \cdot 0,\ 0.1 \cdot 0 + 0.1 \cdot 1 + 0.8 \cdot 0),\ \max(0, 0.5),\ 0, 0, 1, 1\ ]$

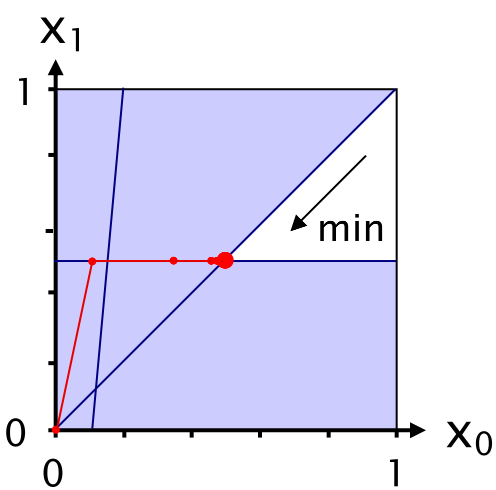
$= [\ 0.1, 0.5, 0, 0, 1, 1\ ]$

$n=2$: $[\ \max(0.6 \cdot 0.5 + 0.4 \cdot 0.1,\ 0.1 \cdot 0.5 + 0.1 \cdot 1 + 0.8 \cdot 0),\ \max(0, 0.5),\ 0, 0, 1, 1\ ]$

$= [\ 0.34, 0.5, 0, 0, 1, 1\ ]$

Example - Reachability (val. iter.)





	$[x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}, x_4^{(n)}, x_5^{(n)}]$
n=0:	[0, 0, 0, 0, 1, 1]
n=1:	[0.1, 0.5, 0, 0, 1, 1]
n=2:	[0.34, 0.5, 0, 0, 1, 1]
n=3:	[0.436, 0.5, 0, 0, 1, 1]
n=4:	[0.4744, 0.5, 0, 0, 1, 1]
n=5:	[0.48976, 0.5, 0, 0, 1, 1]
n=6:	[0.495904, 0.5, 0, 0, 1, 1]
n=7:	[0.4983616, 0.5, 0, 0, 1, 1]
n=8:	[0.49934464, 0.5, 0, 0, 1, 1]
	…
n=16:	[0.49999957, 0.5, 0, 0, 1, 1]
n=17:	[0.49999982, 0.5, 0, 0, 1, 1]
…	≈ [0.5 0.5, 0, 0, 1, 1]

Strategy synthesis

- Compute optimal probabilities $p_{max}(s, F\ b)$ for all $s \in S$
- To compute the optimal strategy σ*, choose the locally optimal action in each state
  - must also guarantee reachability (graph-based)
  - in general depends on the method used to compute the optimal probabilities
  - policy iteration computes the optimal strategy
- For reachability
  - memoryless strategies suffice
- For step-bounded reachability
  - need finite-memory strategies
  - typically requires backward computation from the goal states for a fixed number of steps

Memoryless strategies

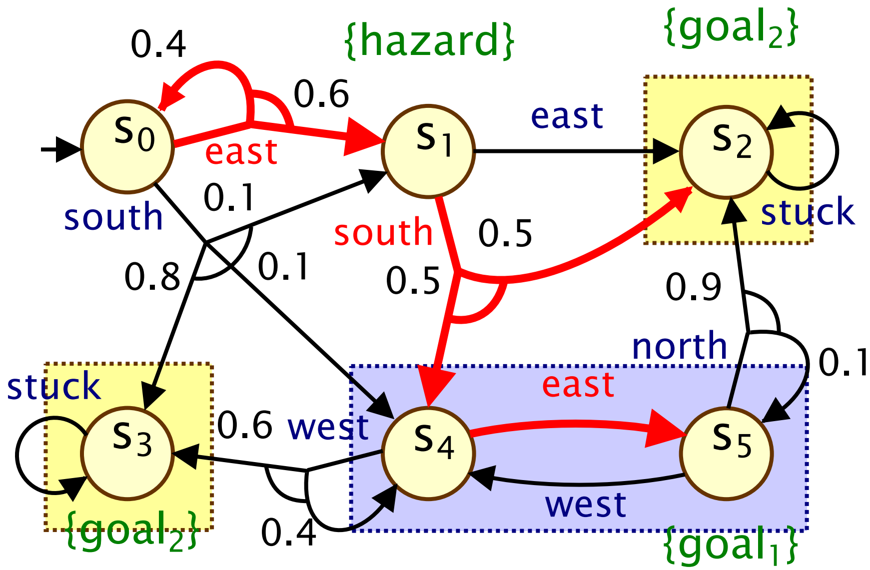
- Memoryless strategies suffice for probabilistic reachability
  - i.e. there exist memoryless strategies σ_{min} & σ_{max} such that:
  - $Prob^{\sigma_{min}}(s, F\ a) = p_{min}(s, F\ a)$ for all states $s \in S$
  - $Prob^{\sigma_{max}}(s, F\ a) = p_{max}(s, F\ a)$ for all states $s \in S$

- Construct strategies from optimal solution:

$$\sigma_{min}(s) = \text{argmin}\left\{ \sum_{s' \in S} \mu(s') \cdot p_{min}(s', F\ a) \mid (a, \mu) \in \textbf{Steps}(s) \right\}$$

$$\sigma_{max}(s) = \text{argmax}\left\{ \sum_{s' \in S} \mu(s') \cdot p_{max}(s', F\ a) \mid (a, \mu) \in \textbf{Steps}(s) \right\}$$

Example – Strategy



Optimal strategy:

s_0 : east

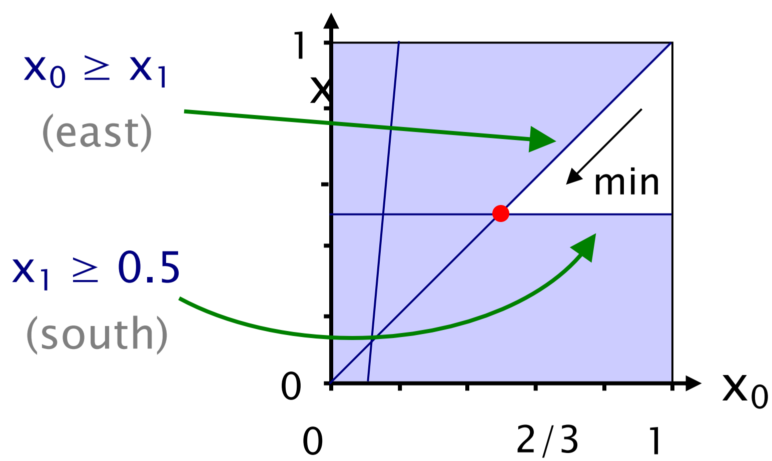
s_1 : south

s_2 : –

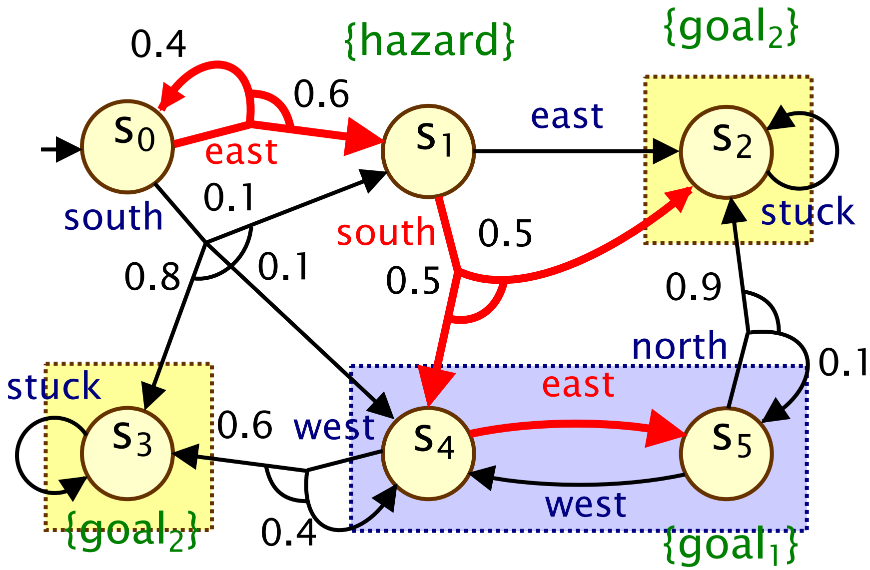
s_3 : –

s_4 : east

s_5 : –



Example - Strategy



Optimal strategy:

s_0 : east

s_1 : south

s_2 : –

s_3 : –

s_4 : east

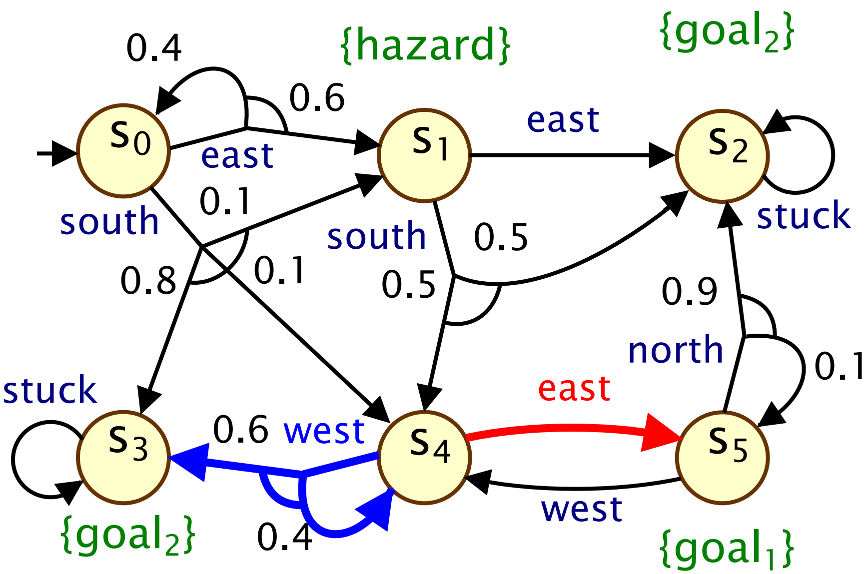
s_5 : –

$\underline{p}_{\underline{max}}(F\ goal_1)$

$= [\ 0.5,\ 0.5,\ 0,\ 0,\ 1,\ 1\ ]$

$= [\ \max(0.6{\cdot}0.5+0.4{\cdot}0.5,\ 0.1{\cdot}0.5+0.1{\cdot}1+0.8{\cdot}0),\ \max(0,\ 0.5),\ 0,\ 0,\ 1,\ 1\ ]$

Example – Bounded reachability



Example:

$P_{max=?}$ [$F^{\leq 3}$ goal$_2$]

So compute:

$p_{max}(s_0, F^{\leq 3}\ goal_2) = 0.99$

Optimal strategy is finite-memory:

s_4 (after 1 step): east

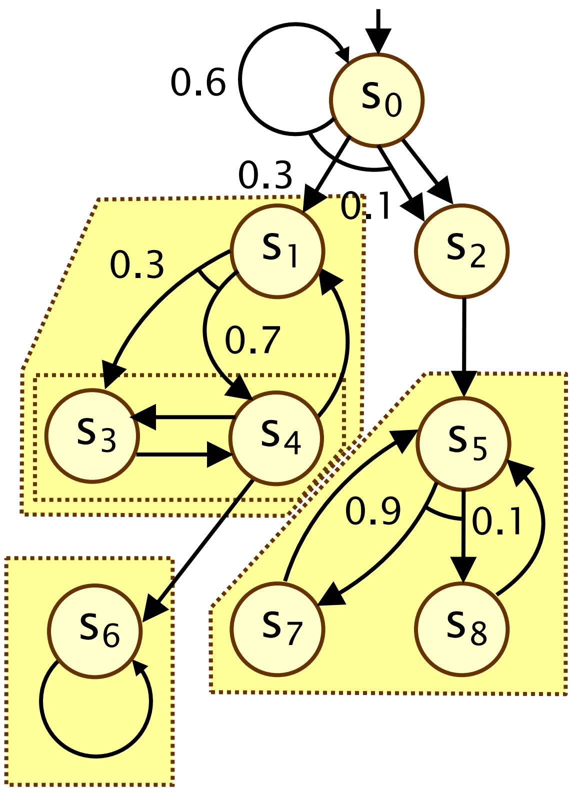
s_4 (after 2 steps): west

Recall – end components in MDPs

- End components of MDPs
  are the analogue of BSCCs in DTMCs

- An end component is a
  strongly connected sub-MDP

- A sub-MDP comprises a subset
  of states and a subset of the
  actions/distributions available
  in those states, which is closed
  under probabilistic branching



Note:
- action labels omitted
- probabilities omitted where =1

Repeated reachability + persistence

- Maximum probabilities
  - $p_{max}(s, GF\ a) = p_{max}(s, F\ T_{GFa})$
    - where T_{GFa} is the union of sets T for all end components (T,**Steps'**) with $T \cap Sat(a) \neq \varnothing$
  - $p_{max}(s, FG\ a) = p_{max}(s, F\ T_{FGa})$
    - where T_{FGa} is the union of sets T for all end components (T,**Steps'**) with $T \subseteq Sat(a)$
- Minimum probabilities
  - need to compute from maximum probabilities…
  - $p_{min}(s, GF\ a) = 1 - p_{max}(s, FG\neg a)$
  - $p_{min}(s, FG\ a) = 1 - p_{max}(s, GF\neg a)$

Automata-based properties for MDPs

- For an MDP M and automaton A over alphabet 2^{AP}
  - consider probability of "satisfying" language $L(A) \subseteq (2^{AP})^\omega$
  - $Prob^{M,\sigma}(s, A) = Pr_s^{M,\sigma}\{ \omega \in Path^{M,\sigma}(s) \mid trace(\omega) \in L(A) \}$
  - $p_{max}^M(s, A) = \sup_{\sigma \in Adv} Prob^{M,\sigma}(s, A)$
  - $p_{min}^M(s, A) = \inf_{\sigma \in Adv} Prob^{M,\sigma}(s, A)$

- Verification might need minimum or maximum probabilities
  - e.g. $s \vDash P_{\geq 0.99} [\ \psi_{good}\ ] \Leftrightarrow p_{min}^M (s, \psi_{good}) \geq 0.99$
  - e.g. $s \vDash P_{\leq 0.05} [\ \psi_{bad}\ ] \Leftrightarrow p_{max}^M (s, \psi_{bad}) \leq 0.05$
- But, ω-regular properties are closed under negation
  - as are the automata that represent them
  - so can always consider maximum probabilities…
  - $p_{max}^M(s, \psi_{bad})$ or $1 - p_{min}^M(s, \psi_{good})$

Automata-based properties for MDPs

- For an MDP M and automaton A over alphabet 2^{AP}
  - consider probability of "satisfying" language $L(A) \subseteq (2^{AP})^\omega$
  - $Prob^{M,\sigma}(s, A) = Pr_s^{M,\sigma}\{ \omega \in Path^{M,\sigma}(s) \mid trace(\omega) \in L(A) \}$
  - $p_{max}^M(s, A) = \sup_{\sigma \in Adv} Prob^{M,\sigma}(s, A)$
  - $p_{min}^M(s, A) = \inf_{\sigma \in Adv} Prob^{M,\sigma}(s, A)$

- Synthesis might need minimum or maximum probabilities
  - Synth strat such that $P_{\geq 0.99} [\ \psi_{good}\ ] \Leftrightarrow p_{max}^M (s, \psi_{good}) \geq 0.99$
  - Synth strat such that $P_{\leq 0.05} [\ \psi_{bad}\ ] \Leftrightarrow p_{min}^M (s, \psi_{bad}) \leq 0.05$
- But, ω-regular properties are closed under negation
  - as are the automata that represent them
  - so can always consider maximum probabilities…
  - $p_{max}^M(s, \psi_{good})$ or $1 - p_{min}^M(s, \psi_{bad})$

LTL strategy synthesis for MDPs (max)

- Synthesise strategy σ over MDP M such that $Prob^{M,\sigma}(s, \psi) = p_{max}^{M}(s, \psi)$
- 1. Generate a DRA for ψ
  - build nondeterministic Büchi automaton (NBA) for ψ [VW94]
  - convert the NBA to a DRA [Saf88] M
- 2. Construct product MDP $M \otimes A$
- 3. Identify accepting end components (ECs) of $M \otimes A$
- 4. Compute max probability of reaching accepting ECs
  - from all states of the $M \otimes A$
- 5. Check if probability for (s, q_s) against p for each s

Product MDP for a DRA

- For an MDP $M = (S, s_{init}, \mathbf{Steps}, L)$
- and a (total) DRA $A = (Q, \Sigma, \delta, q_0, Acc)$
  - where $Acc = \{ (L_i, K_i) \mid 1 \leq i \leq k \}$

- The product MDP $M \otimes A$ is:
  - the MDP $(S \times Q, (s_{init}, q_{init}), \mathbf{Steps}', L')$ where:

$$q_{init} = \delta(q_0, L(s_{init}))$$

$$\mathbf{Steps}'((s,q)) = \{ \mu^q \mid \mu \in Step(s) \}$$

$$\mu^q(s',q') = \begin{cases} \mu(s') & \text{if } q' = \delta(q, L(s)) \\ 0 & \text{otherwise} \end{cases}$$

$l_i \in L'((s,q))$ if $q \in L_i$ and $k_i \in L'((s,q))$ if $q \in K_i$

(i.e. state sets of acceptance condition used as labels)

Product MDP for a DRA

- For MDP M and DRA A

$$p_{max}^{M}(s, A) = p_{max}^{M\otimes A}((s,q_s), \vee_{1 \leq i \leq k} (FG\ \neg l_i \wedge GF\ k_i))$$

  - where $q_s = \delta(q_0, L(s))$

- Hence:

$$p_{max}^{M}(s, A) = p_{max}^{M\otimes A}((s,q_s), F\ T_{Acc})$$

  - where T_{Acc} is the union of all sets T for accepting end components (T,**Steps'**) in D⊗A
  - an accepting end component is such that, for some $1 \leq i \leq k$:
    - $(s,q) \models \neg l_i$ for all $(s,q) \in T$ and $(s,q) \models k_i$ for some $(s,q) \in T$
    - i.e. $T \cap (S \times L_i) = \emptyset$ and $T \cap (S \times K_i) \neq \emptyset$

Strategy synthesis for LTL objectives

- Reduce to a reachability problem on the product of MDP M and an ω-automaton representing ψ
  - for example, deterministic Rabin automaton (DRA)

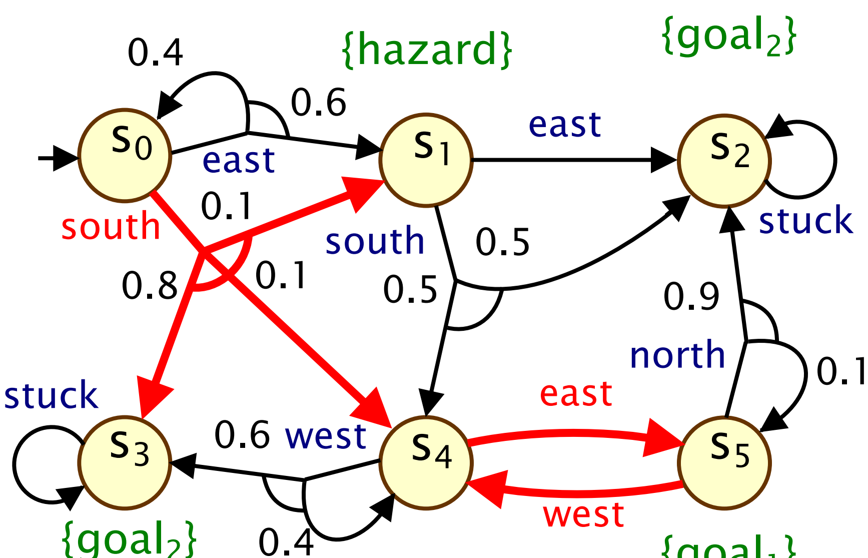
- Need only consider computation of maximum probabilities $p_{max}(s,\psi)$
  - since $p_{min}(s,\psi) = 1 - p_{max}(s,\neg\psi)$

- To compute the optimal strategy σ*
  - find memoryless strategy on the product MDP
  - convert to finite-memory strategy with one mode for each state of the DRA for ψ

Example – LTL

- $P_{\geq 0.05}$ [(G ¬hazard) ∧ (GF $goal_1$)]
  – avoid hazard and visit $goal_1$ infinitely often

- $p_{max}(s_0$, (G ¬hazard) ∧ (GF $goal_1$)) = 0.1



Optimal strategy:
(in this instance,
memoryless)

s_0 : south
s_1 : –
s_2 : –
s_3 : –
s_4 : east
s_5 : west

Strategy synthesis for reward properties

- Cumulative: $R_{\sim r}$ [$C^{\leq k}$]
  - similar to step-bounded probabilistic reachability
  - optimal strategies are deterministic but may need finite memory
  - solution of recursive equations, with k iterations
  - $R_{\sim r}$ [C] is the total expected reward, more complex…

See [FKNP11] for details

- Reachability: $R_{\sim r}$ [F ϕ]
  - similar to the case of probabilistic reachability
  - precomputation to identify states that do not reach ϕ, assigned infinite rewards
  - solve a linear optimization problem (or value iteration)
  - optimal strategies are memoryless deterministic

Summing up…

- The strategy synthesis problem
  - solved using the same methods as the verification problem
  - extract optimal strategy/policy/adversary
  - correct-by-construction synthesis procedure
- Reward properties
  - can be handled similarly