Rich Behavioral Models: Illustration on Journey Planning and Focus on Multi-Constraint Percentiles Queries in MDPs

Mickael Randour

Computer Science Department, ULB - Université libre de Bruxelles, Belgium

March 20, 2017

Informatik Kolloquium — RWTH Aachen

The talk in one slide

Strategy synthesis for Markov Decision Processes (MDPs)

Finding **good** controllers for systems interacting with a *stochastic* environment.

The talk in one slide

Strategy synthesis for Markov Decision Processes (MDPs)

Finding **good** controllers for systems interacting with a *stochastic* environment.

- Good? Performance evaluated through *payoff functions*.
- Usual problem is to optimize the *expected performance* or the *probability of achieving a given performance level*.
- Not sufficient for many practical applications.
  - ▷ Several extensions, more expressive but also more complex...

The talk in one slide

Strategy synthesis for Markov Decision Processes (MDPs)

Finding **good** controllers for systems interacting with a *stochastic* environment.

- Good? Performance evaluated through *payoff functions*.
- Usual problem is to optimize the *expected performance* or the *probability of achieving a given performance level*.
- Not sufficient for many practical applications.
  - ▷ Several extensions, more expressive but also more complex...

Aim of this survey talk

Give a flavor of classical questions and extensions (*rich behavioral models*), illustrated on the stochastic shortest path (SSP).

The talk in one slide

Strategy synthesis for Markov Decision Processes (MDPs)

Finding **good** controllers for systems interacting with a *stochastic* environment.

- Good? Performance evaluated through *payoff functions*.
- Usual problem is to optimize the *expected performance* or the *probability of achieving a given performance level*.
- Not sufficient for many practical applications.
  - ▷ Several extensions, more expressive but also more complex...

Aim of this survey talk

Give a flavor of classical questions and extensions (*rich behavioral models*), illustrated on the stochastic shortest path (SSP).

+ *Brief focus on percentile queries.*

1 Context, MDPs, strategies

2 Classical stochastic shortest path problems

3 Good expectation under acceptable worst-case

4 Percentile queries in multi-dimensional MDPs

5 Conclusion

Multi-criteria quantitative synthesis

- Verification and synthesis:
  - a reactive **system** to *control*,
  - an *interacting* **environment**,
  - a **specification** to *enforce*.

Multi-criteria quantitative synthesis

- Verification and synthesis:
  - a reactive **system** to *control*,
  - an *interacting* **environment**,
  - a **specification** to *enforce*.
- Model of the (discrete) interaction?
  - Antagonistic environment: 2-player game on graph.
  - **Stochastic environment: MDP**.

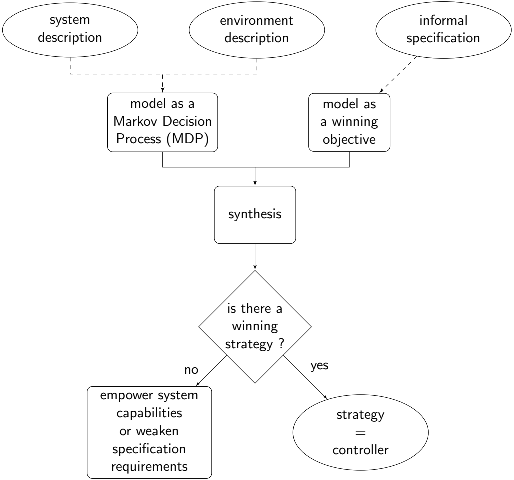
Multi-criteria quantitative synthesis

- Verification and synthesis:
  - ▷ a reactive **system** to *control*,
  - ▷ an *interacting* **environment**,
  - ▷ a **specification** to *enforce*.
- Model of the (discrete) interaction?
  - ▷ Antagonistic environment: 2-player game on graph.
  - ▷ **Stochastic environment: MDP**.
- **Quantitative** specifications. Examples:
  - ▷ Reach a state s before x time units $\rightsquigarrow$ shortest path.
  - ▷ Minimize the average response-time $\rightsquigarrow$ mean-payoff.

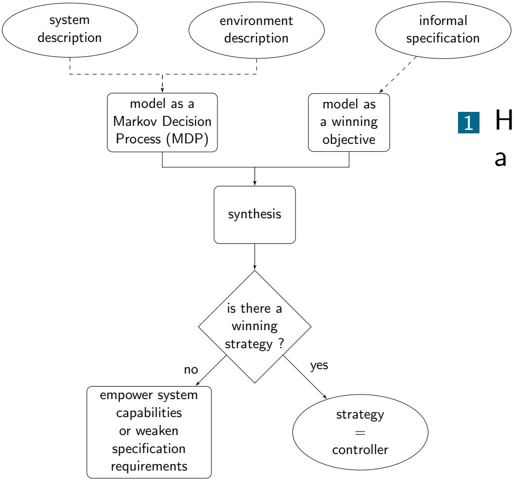
Multi-criteria quantitative synthesis

- Verification and synthesis:
  - a reactive **system** to *control*,
  - an *interacting* **environment**,
  - a **specification** to *enforce*.
- Model of the (discrete) interaction?
  - Antagonistic environment: 2-player game on graph.
  - **Stochastic environment: MDP**.
- **Quantitative** specifications. Examples:
  - Reach a state s before x time units $\leadsto$ shortest path.
  - Minimize the average response-time $\leadsto$ mean-payoff.
- Focus on **multi-criteria quantitative models**
  - to reason about *trade-offs* and *interplays*.

Strategy (policy) synthesis for MDPs

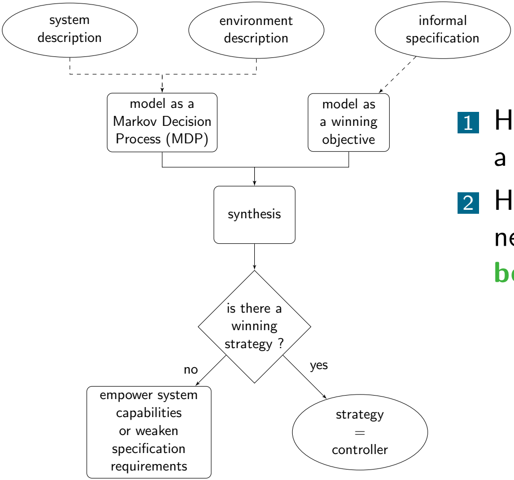


Strategy (policy) synthesis for MDPs



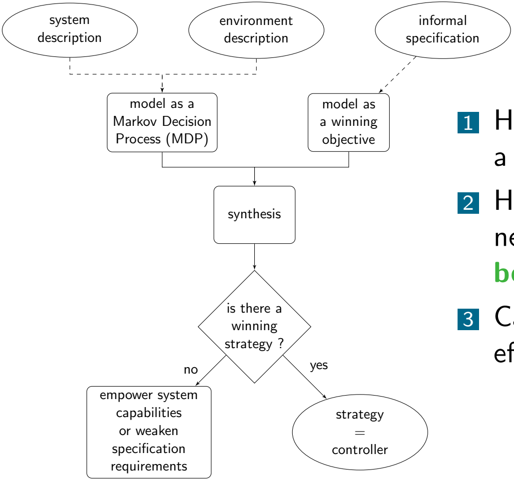
1. How complex is it to decide if a winning strategy exists?

Strategy (policy) synthesis for MDPs



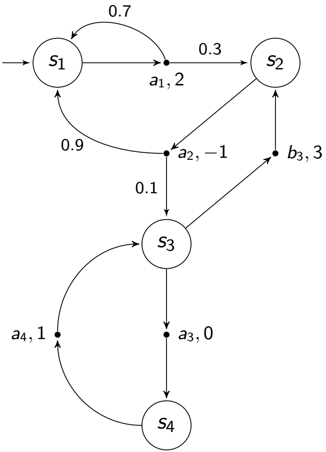
1. How complex is it to decide if a winning strategy exists?
2. How complex such a strategy needs to be? **Simpler is better**.

Strategy (policy) synthesis for MDPs



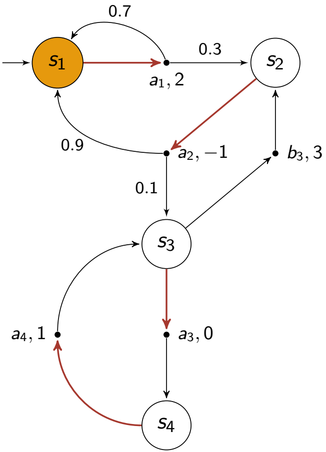
1. How complex is it to decide if a winning strategy exists?
2. How complex such a strategy needs to be? **Simpler is better.**
3. Can we synthesize one efficiently?

Markov decision processes



- MDP $D = (S, s_{\mathsf{init}}, A, \delta, w)$.
  - ▷ Finite sets of states S and actions A,
  - ▷ probabilistic transition $\delta\colon S \times A \rightarrow \mathcal{D}(S)$,
  - ▷ weight function $w\colon A \rightarrow \mathbb{Z}$.
- **Run** (or play): $\rho = s_1 a_1 \ldots a_{n-1} s_n \ldots$ such that $\delta(s_i, a_i, s_{i+1}) > 0$ for all $i \geq 1$.
  - ▷ Set of runs $\mathcal{R}(D)$.
  - ▷ Set of histories (finite runs) $\mathcal{H}(D)$.
- **Strategy** $\sigma\colon \mathcal{H}(D) \rightarrow \mathcal{D}(A)$.
  - ▷ $\forall\, h$ ending in s, $\mathsf{Supp}(\sigma(h)) \in A(s)$.

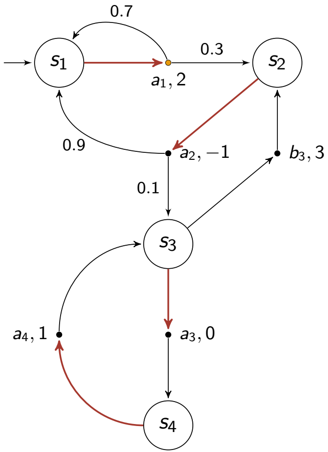
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1$

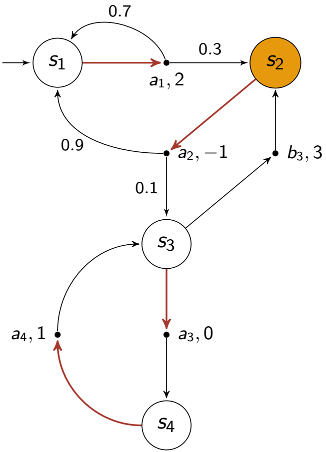
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1$

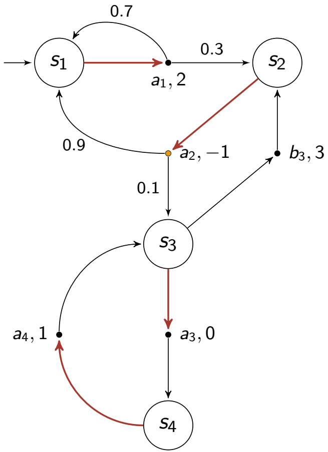
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2$

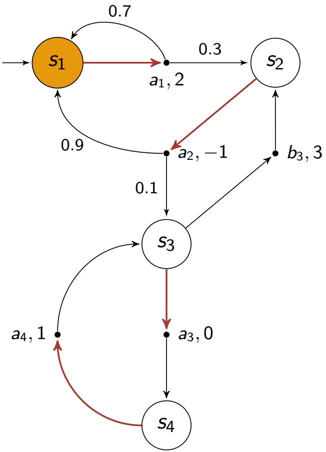
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2$

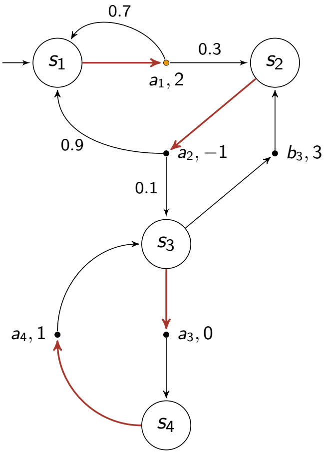
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1$

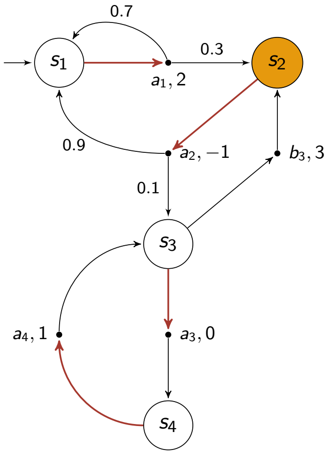
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1$

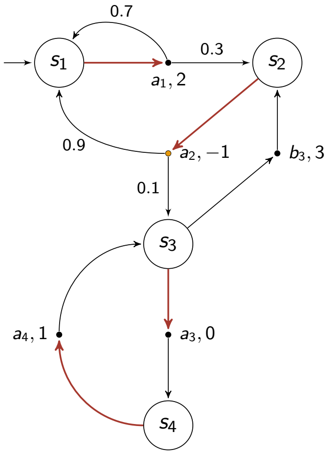
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2$

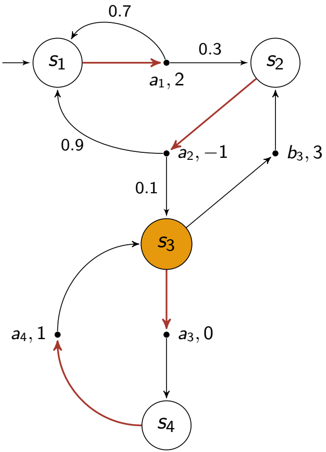
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2 a_2$

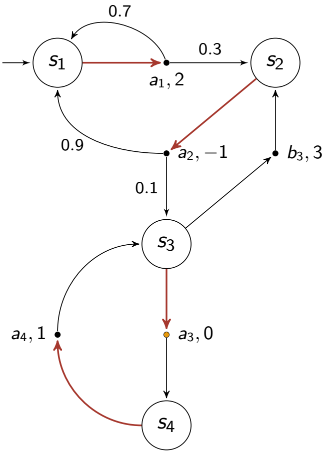
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2 a_2 s_3$

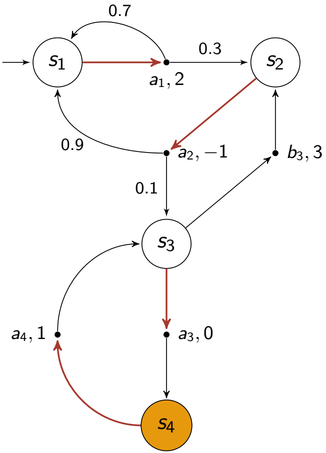
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2 a_2 s_3 a_3$

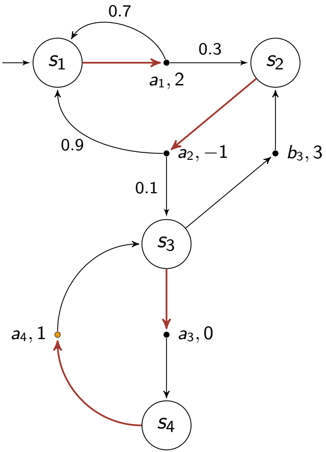
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2 a_2 s_3 a_3 s_4$

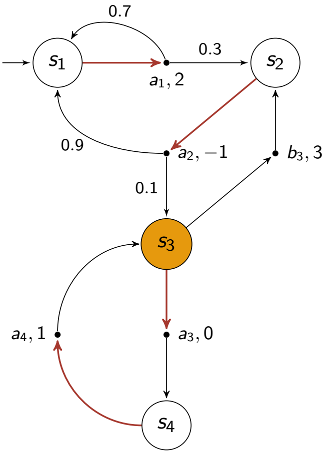
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2 a_2 s_3 a_3 s_4 a_4$

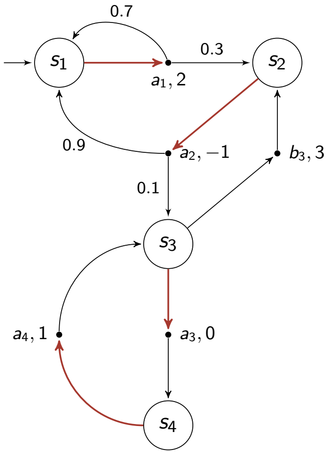
Markov decision processes



Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2 a_2 (s_3 a_3 s_4 a_4)^\omega$.

Markov decision processes

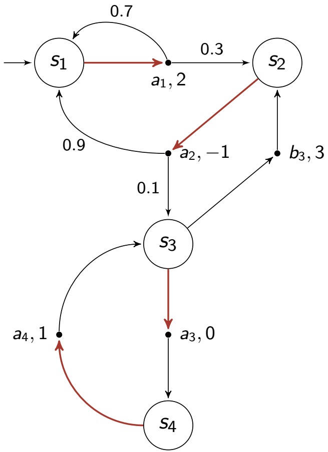


Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2 a_2 (s_3 a_3 s_4 a_4)^\omega$.

Other possible run $\rho' = s_1 a_1 s_2 a_2 (s_3 a_3 s_4 a_4)^\omega$.

Markov decision processes



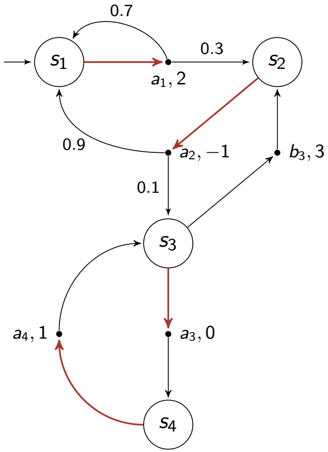
Sample *pure memoryless* strategy σ.

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2 a_2 (s_3 a_3 s_4 a_4)^\omega$.

Other possible run $\rho' = s_1 a_1 s_2 a_2 (s_3 a_3 s_4 a_4)^\omega$.

- Strategies may use
  - finite or infinite **memory**,
  - **randomness**.
- **Payoff functions** map runs to numerical values:
  - truncated sum up to $T = \{s_3\}$: $\mathsf{TS}^T(\rho) = 2$, $\mathsf{TS}^T(\rho') = 1$,
  - mean-payoff: $\underline{\mathsf{MP}}(\rho) = \underline{\mathsf{MP}}(\rho') = 1/2$,
  - many more.

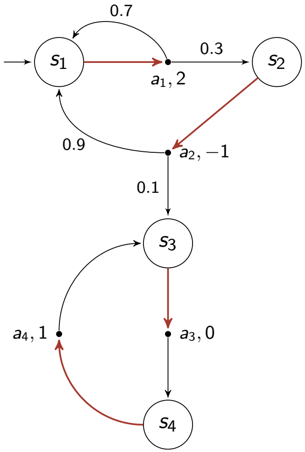
Markov chains



Once strategy σ fixed, fully stochastic process:

$\rightsquigarrow$ **Markov chain** (**MC**) M.

Markov chains

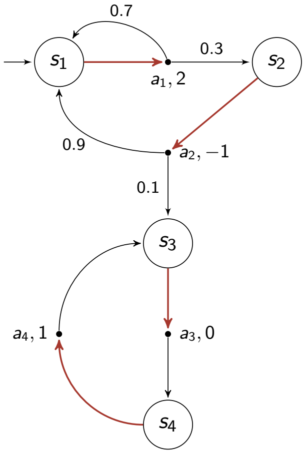


Once strategy σ fixed, fully stochastic process:

$\rightsquigarrow$ **Markov chain** (**MC**) M.

State space = product of the MDP and the memory of σ.

Markov chains



Once strategy σ fixed, fully stochastic process:

$\rightsquigarrow$ **Markov chain** (**MC**) M.

State space = product of the MDP and the memory of σ.

- Event $\mathcal{E} \subseteq \mathcal{R}(M)$
  - ▷ probability $\mathbb{P}_M(\mathcal{E})$
- Measurable $f\colon \mathcal{R}(M) \to \mathbb{R} \cup \{\infty\}$,
  - ▷ expected value $\mathbb{E}_M(f)$

Aim of this survey

Compare different types of quantitative specifications for MDPs

- w.r.t. the complexity of the decision problem,
- w.r.t. the complexity of winning strategies.

Aim of this survey

Compare different types of quantitative specifications for MDPs

- w.r.t. the complexity of the decision problem,
- w.r.t. the complexity of winning strategies.

Recent extensions share a common philosophy: framework for the synthesis of strategies with *richer performance guarantees*.

- Our work deals with many different payoff functions.

Aim of this survey

Compare different types of quantitative specifications for MDPs

- ▷ w.r.t. the complexity of the decision problem,
- ▷ w.r.t. the complexity of winning strategies.

Recent extensions share a common philosophy: framework for the synthesis of strategies with *richer performance guarantees*.

- ▷ Our work deals with many different payoff functions.

Focus on the **shortest path problem** in this talk.

- ▷ Not the most involved technically, natural applications.
- ⇝ Useful to understand the practical interest of each variant.
  - + Brief mention of results for other payoffs.

Aim of this survey

Compare different types of quantitative specifications for MDPs

- ▷ w.r.t. the complexity of the decision problem,
- ▷ w.r.t. the complexity of winning strategies.

Recent extensions share a common philosophy: framework for the synthesis of strategies with *richer performance guarantees*.

- ▷ Our work deals with many different payoff functions.

Focus on the **shortest path problem** in this talk.

- ▷ Not the most involved technically, natural applications.
- ⇝ Useful to understand the practical interest of each variant.
  - + Brief mention of results for other payoffs.

Based on joint work with R. Berthon, V. Bruyère, E. Filiot, J.-F. Raskin, O. Sankur [BFRR14b, BFRR14a, RRS15a, RRS15b, BCH^{+}16, Ran16, BRR17].

Stochastic shortest path

Shortest path problem for *weighted graphs*

Given state $s \in S$ and target set $T \subseteq S$, find a path from s to a state $t \in T$ that minimizes the sum of weights along edges.

- PTIME algorithms (Dijkstra, Bellman-Ford, etc) [CGR96].

Stochastic shortest path

Shortest path problem for *weighted graphs*

Given state $s \in S$ and target set $T \subseteq S$, find a path from s to a state $t \in T$ that minimizes the sum of weights along edges.

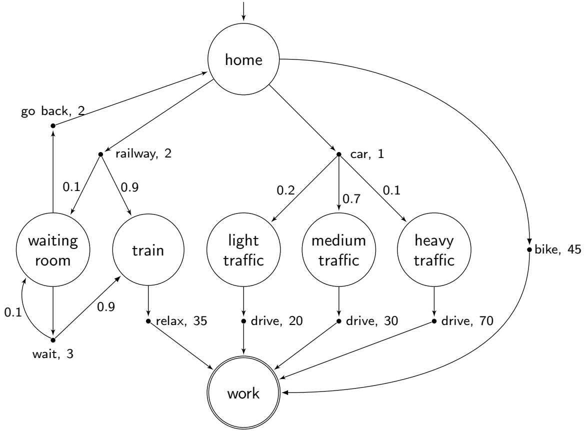
- PTIME algorithms (Dijkstra, Bellman-Ford, etc) [CGR96].

We focus on MDPs with **strictly positive weights** for the SSP.

- **Truncated sum** payoff function for $\rho = s_1 a_1 s_2 a_2 \dots$ and target set T:

$$\mathsf{TS}^T(\rho) = \begin{cases} \sum_{j=1}^{n-1} w(a_j) \text{ if } s_n \text{ first visit of } T, \\ \infty \text{ if } T \text{ is never reached.} \end{cases}$$

Planning a journey in an uncertain environment



Each action takes time, target = work.

- ▷ What kind of strategies are we looking for when the environment is stochastic?

SSP-E: minimizing the expected length to target

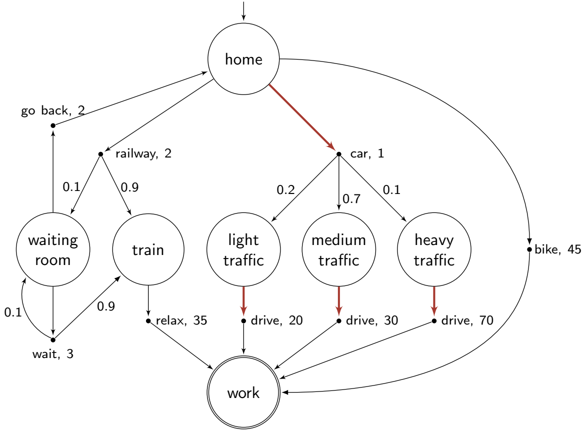
SSP-E problem

Given MDP $D = (S, s_{\text{init}}, A, \delta, w)$, target set T and threshold $\ell \in \mathbb{Q}$, decide if there exists σ such that $\mathbb{E}^{\sigma}_{D}(\mathsf{TS}^{T}) \leq \ell$.

Theorem [BT91]

The SSP-E problem can be decided in polynomial time. Optimal pure memoryless strategies always exist and can be constructed in polynomial time.

SSP-E: illustration



- ▷ Pure memoryless strategies suffice.
- ▷ Taking the **car** is optimal: $\mathbb{E}_D^{\sigma}(\mathsf{TS}^T) = 33$.

SSP-E: PTIME algorithm

1. Graph analysis (linear time):
   - s not connected to $T \Rightarrow \infty$ and remove,
   - $s \in T \Rightarrow 0$.
2. **Linear programming** (**LP**, polynomial time).

SSP-E: PTIME algorithm

1. Graph analysis (linear time):
   - s not connected to $T \Rightarrow \infty$ and remove,
   - $s \in T \Rightarrow 0$.
2. **Linear programming** (**LP**, polynomial time).

For each $s \in S \setminus T$, one variable x_s,

$$\max \sum_{s \in S \setminus T} x_s$$

under the constraints

$$x_s \leq w(a) + \sum_{s' \in S \setminus T} \delta(s, a, s') \cdot x_{s'} \qquad \text{for all } s \in S \setminus T, \text{ for all } a \in A(s).$$

SSP-E: PTIME algorithm

1. Graph analysis (linear time):
   - s not connected to $T \Rightarrow \infty$ and remove,
   - $s \in T \Rightarrow 0$.
2. **Linear programming** (**LP**, polynomial time).

Optimal solution $\mathbf{v}$:

$\rightsquigarrow$ $\mathbf{v}_s$ = expectation from s to T under an optimal strategy.

Optimal pure memoryless strategy $\sigma^{\mathbf{v}}$:

$$\sigma^{\mathbf{v}}(s) = \arg\min_{a \in A(s)} \left[w(a) + \sum_{s' \in S \setminus T} \delta(s, a, s') \cdot \mathbf{v}_{s'} \right].$$

$\rightsquigarrow$ **Playing optimally = locally optimizing present + future.**

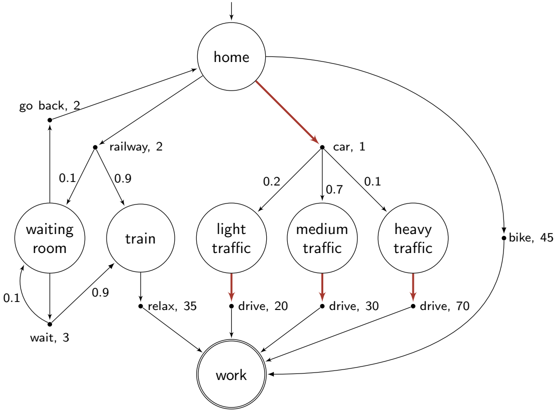
SSP-E: PTIME algorithm

1. Graph analysis (linear time):
   - ▷ s not connected to $T \Rightarrow \infty$ and remove,
   - ▷ $s \in T \Rightarrow 0$.
2. **Linear programming** (**LP**, polynomial time).

In practice, **value and strategy iteration** algorithms often used:

- ▷ best performance in most cases but exponential in the worst-case,
- ▷ fixed point algorithms, successive solution improvements [BT91, dA99, HM14].

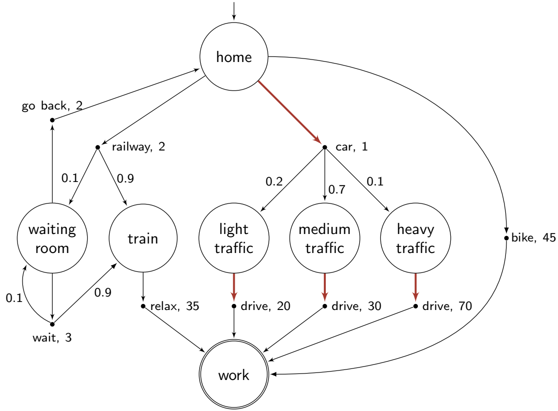
Travelling without taking too many risks



Minimizing the *expected time* to destination makes sense **if** we travel often and it is not a problem to be late.

With car, in 10% of the cases, the journey takes 71 minutes.

Travelling without taking too many risks



Most bosses will not be happy if we are late too often. . .

$\rightsquigarrow$ what if we are risk-averse and want to avoid that?

SSP-P: forcing short paths with high probability

SSP-P problem

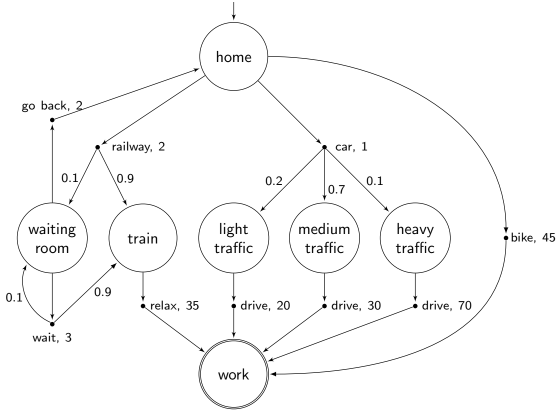
Given MDP $D = (S, s_{\text{init}}, A, \delta, w)$, target set T, threshold $\ell \in \mathbb{N}$, and probability threshold $\alpha \in [0,1] \cap \mathbb{Q}$, decide if there exists a strategy σ such that $\mathbb{P}^{\sigma}_{D}\big[\{\rho \in \mathcal{R}_{s_{\text{init}}}(D) \mid \mathsf{TS}^{T}(\rho) \leq \ell\}\big] \geq \alpha$.

Theorem

The SSP-P problem can be decided in pseudo-polynomial time, and it is PSPACE-hard. Optimal pure strategies with pseudo-polynomial memory always exist and can be constructed in pseudo-polynomial time.

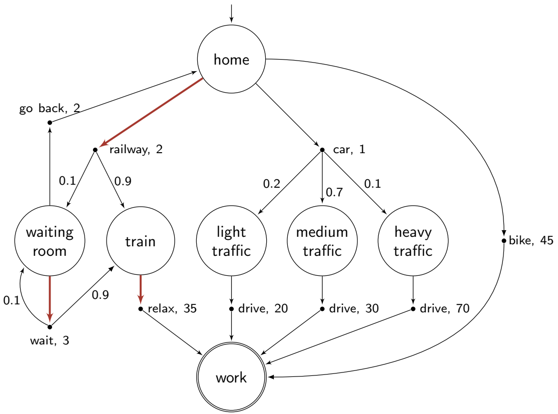
See [HK15] for hardness and for example [RRS15a] for algorithm.

SSP-P: illustration



Specification: reach work within 40 minutes with 0.95 probability

SSP-P: illustration



Specification: reach work within 40 minutes with 0.95 probability

Sample strategy: take the **train** $\rightsquigarrow \mathbb{P}^{\sigma}_{D}\big[\mathsf{TS}^{\mathsf{work}} \leq 40\big] = 0.99$

Bad choices: car (0.9) and bike (0.0)

SSP-P: pseudo-PTIME algorithm (1/2)

Key idea: pseudo-PTIME reduction to the **stochastic reachability problem** (**SR**)

SSP-P: pseudo-PTIME algorithm (1/2)

Key idea: pseudo-PTIME reduction to the **stochastic reachability problem** (**SR**)

SR problem

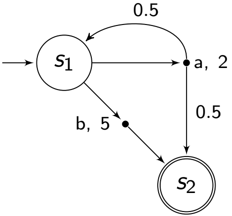
Given unweighted MDP $D = (S, s_{\text{init}}, A, \delta)$, target set T and probability threshold $\alpha \in [0, 1] \cap \mathbb{Q}$, decide if there exists a strategy σ such that $\mathbb{P}^{\sigma}_{D}\left[\Diamond T\right] \geq \alpha$.

Theorem

The SR problem can be decided in polynomial time. Optimal pure memoryless strategies always exist and can be constructed in polynomial time.

- ▷ Linear programming (similar to SSP-E).

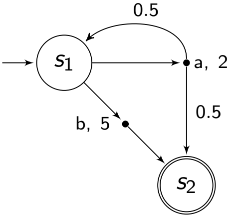
SSP-P: pseudo-PTIME algorithm (2/2)



Sketch of the reduction:

1. Start from D, $T = \{s_2\}$, and $\ell = 7$.

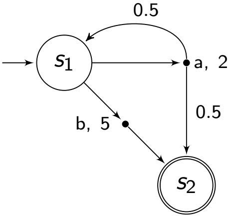
SSP-P: pseudo-PTIME algorithm (2/2)

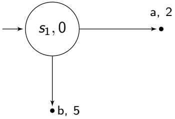


Sketch of the reduction:

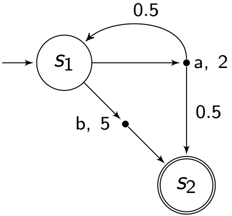
1. Start from D, $T = \{s_2\}$, and $\ell = 7$.
2. Build D_ℓ by unfolding D, tracking the current sum *up to the threshold* ℓ, and integrating it in the states of the expanded MDP.

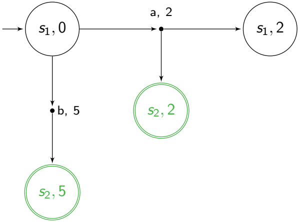
SSP-P: pseudo-PTIME algorithm (2/2)



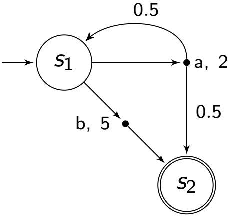


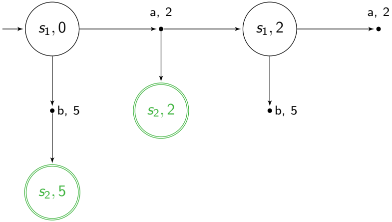
SSP-P: pseudo-PTIME algorithm (2/2)



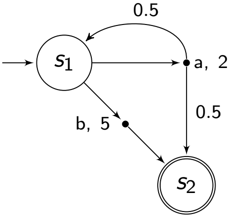


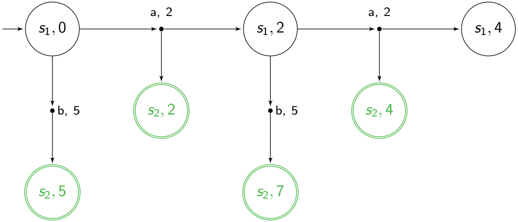
SSP-P: pseudo-PTIME algorithm (2/2)



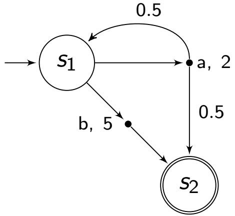


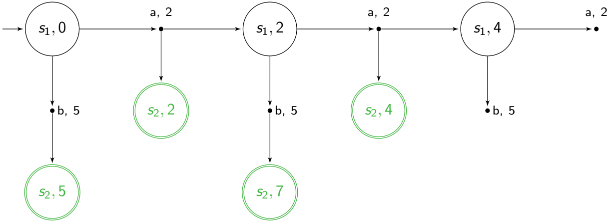
SSP-P: pseudo-PTIME algorithm (2/2)



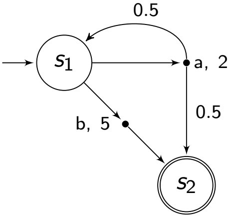


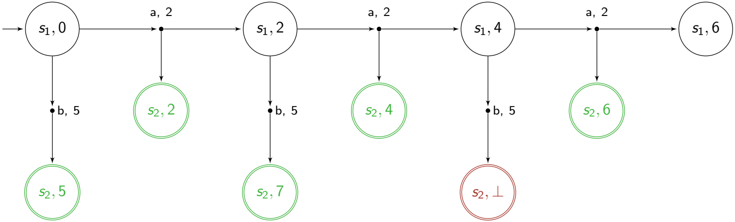
SSP-P: pseudo-PTIME algorithm (2/2)



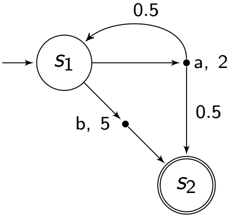


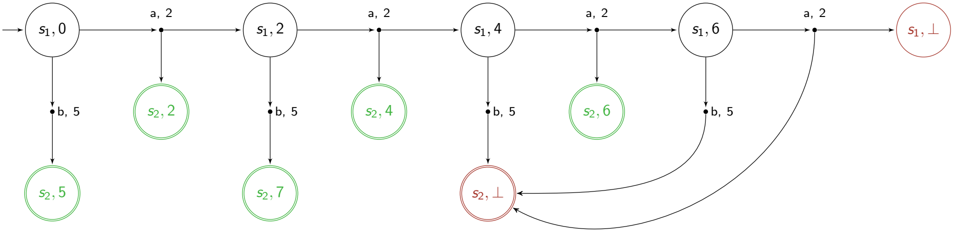
SSP-P: pseudo-PTIME algorithm (2/2)





SSP-P: pseudo-PTIME algorithm (2/2)

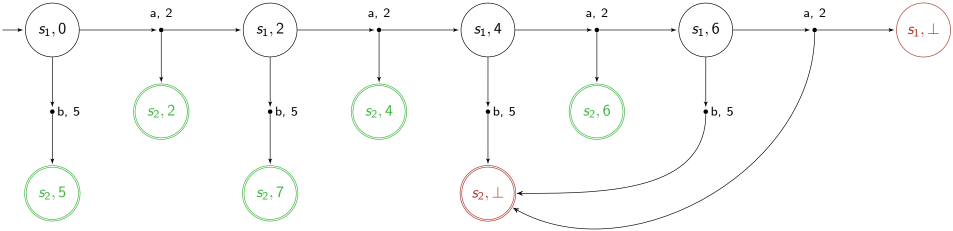




SSP-P: pseudo-PTIME algorithm (2/2)

3. Bijection between runs of D and D_ℓ:

$$\mathsf{TS}^T(\rho) \leq \ell \quad \Leftrightarrow \quad \rho' \models \Diamond T',\ T' = T \times \{0, 1, \ldots, \ell\}.$$



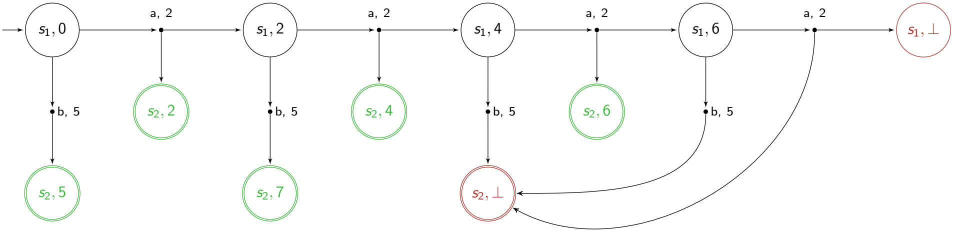
SSP-P: pseudo-PTIME algorithm (2/2)

3 Bijection between runs of D and D_ℓ:

$$\mathsf{TS}^T(\rho) \leq \ell \quad \Leftrightarrow \quad \rho' \models \Diamond T',\ T' = T \times \{0, 1, \ldots, \ell\}.$$

4 Solve the SR problem on D_ℓ.

- ▷ Memoryless strategy in $D_\ell \rightsquigarrow$ pseudo-polynomial memory in D in general.



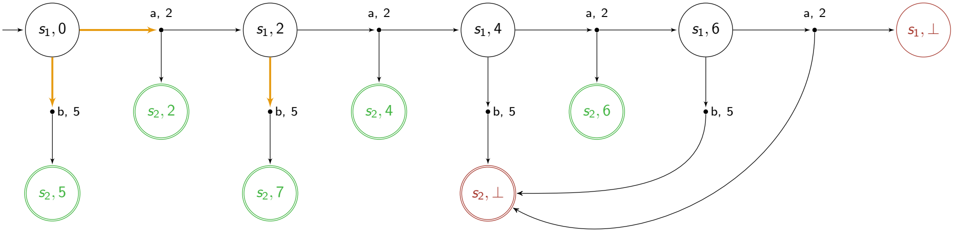
SSP-P: pseudo-PTIME algorithm (2/2)

If we just want to minimize the risk of exceeding $\ell = 7$,

- ▷ an obvious possibility is to play *b* directly,
- ▷ playing *a* only once is also acceptable.

For the SSP-P problem, **both strategies are equivalent**.

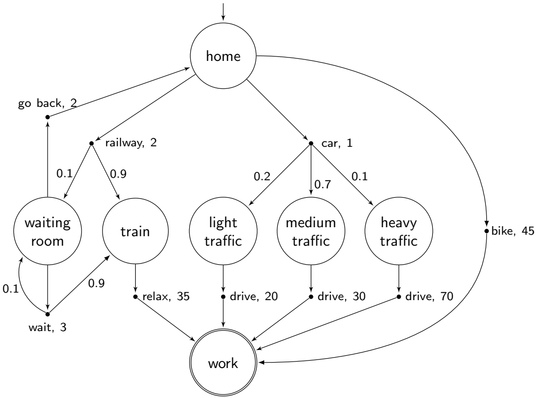
$\rightsquigarrow$ We need richer models to discriminate them!



Related work (non-exhaustive)

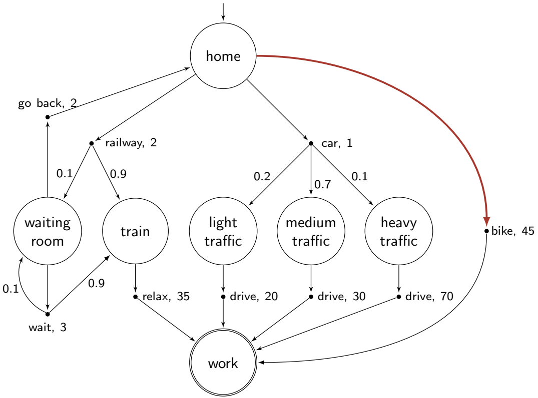
- SSP-P problem [Oht04, SO13].
- *Quantile queries* [UB13]: minimizing the value ℓ of an SSP-P problem for some fixed α. Recently extended to *cost problems* [HK15].
- SSP-E problem in **multi-dimensional** MDPs [FKN$^+$11].

SP-G: strict worst-case guarantees



Specification: *guarantee* that work is reached within 60 minutes (to avoid missing an important meeting).

SP-G: strict worst-case guarantees

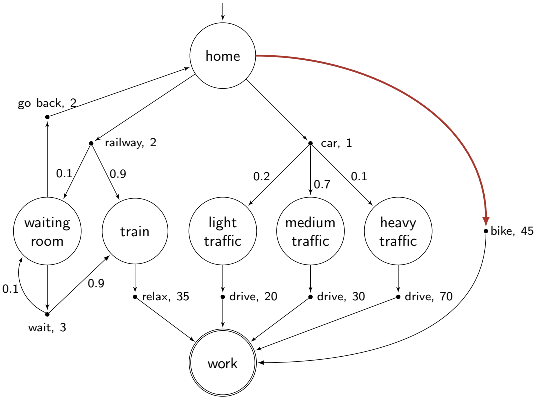


Specification: *guarantee* that work is reached within 60 minutes (to avoid missing an important meeting).

Sample strategy: take the **bike** $\rightsquigarrow \forall\, \rho \in \mathrm{Out}^{\sigma}_{D}\colon\ \mathsf{TS}^{\mathsf{work}}(\rho) \leq 60$.

Bad choices: train ($wc = \infty$) and car ($wc = 71$).

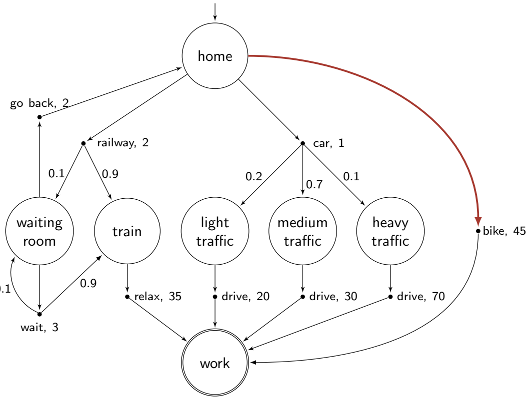
SP-G: strict worst-case guarantees



Winning surely (worst-case) $\neq$ almost-surely (proba. 1).

▷ Train ensures reaching work with probability one, but does not prevent runs where work is never reached.

SP-G: strict worst-case guarantees



Worst-case analysis $\rightsquigarrow$ **two-player game** against an antagonistic adversary.

▷ Forget about probabilities and give the choice of transitions to the adversary.

SP-G: shortest path game problem

SP-G problem

Given MDP $D = (S, s_{\text{init}}, A, \delta, w)$, target set T and threshold $\ell \in \mathbb{N}$, decide if there exists a strategy σ such that for all $\rho \in \text{Out}_D^{\sigma}$, we have that $\text{TS}^T(\rho) \leq \ell$.

Theorem [KBB+08]

The SP-G problem can be decided in polynomial time. Optimal pure memoryless strategies always exist and can be constructed in polynomial time.

- ▷ Does not hold for arbitrary weights.

SP-G: PTIME algorithm

1. Cycles are bad $\implies$ must reach target within $n = |S|$ steps.
2. $\forall\, s \in S$, $\forall\, i$, $0 \leq i \leq n$, compute $\mathbb{C}(s, i)$.
   - Lowest bound on cost to T from s that we can ensure in i steps.
   - **Dynamic programming** (polynomial time).

   Initialize

   $$\forall\, s \in T,\ \mathbb{C}(s, 0) = 0, \qquad \forall\, s \in S \setminus T,\ \mathbb{C}(s, 0) = \infty.$$

   Then, $\forall\, s \in S$, $\forall\, i$, $1 \leq i \leq n$,

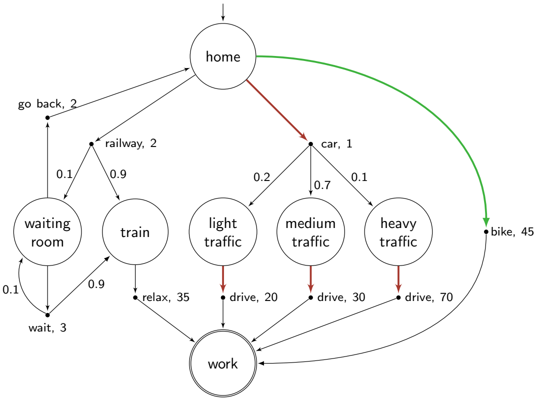
   $$\mathbb{C}(s, i) = \min \left[\mathbb{C}(s, i-1),\ \min_{a \in A(s)}\ \max_{s' \in \mathrm{Supp}(\delta(s,a))} w(a) + \mathbb{C}(s', i-1) \right].$$

3. Winning strategy iff $\mathbb{C}(s_{\mathsf{init}}, n) \leq \ell$.

Related work (non-exhaustive)

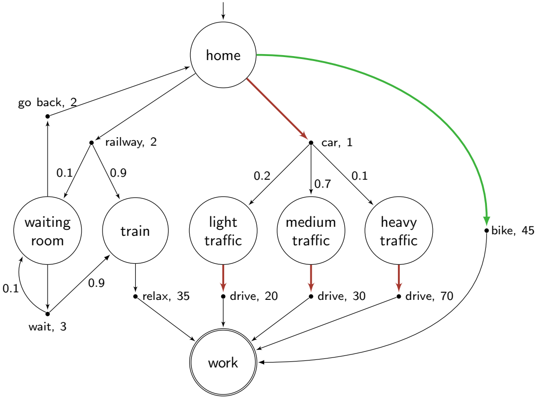
- Pseudo-PTIME for arbitrary weights [BGHM17, FGR15].
- Arbitrary weights + multiple dimensions $\rightsquigarrow$ undecidable (by adapting the proof of [CDRR15] for total-payoff).

SSP-WE $=$ SP-G $\cap$ SSP-E - illustration



- SSP-E: car $\rightsquigarrow \mathbb{E} = 33$ but $wc = 71 > 60$
- SP-G: bike $\rightsquigarrow wc = 45 < 60$ but $\mathbb{E} = 45 >>> 33$

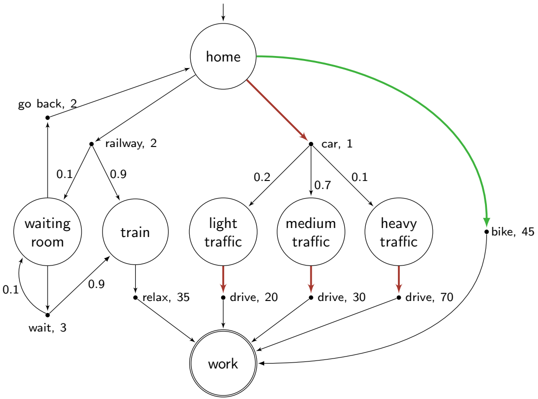
SSP-WE = SP-G ∩ SSP-E - illustration



Can we do better?

▷ **Beyond worst-case synthesis** [BFRR14b, BFRR14a]: minimize the expected time under the worst-case constraint.

SSP-WE = SP-G ∩ SSP-E - illustration



Sample strategy: try train up to 3 delays then switch to bike.

$\rightsquigarrow$ $wc = 58 < 60$ and $\mathbb{E} \approx 37.34 << 45$

$\rightsquigarrow$ pure *finite-memory* strategy

SSP-WE: beyond worst-case synthesis

SSP-WE problem

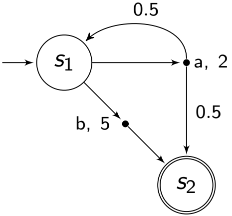
Given MDP $D = (S, s_{\text{init}}, A, \delta, w)$, target set T, and thresholds $\ell_1 \in \mathbb{N}$, $\ell_2 \in \mathbb{Q}$, decide if there exists a strategy σ such that:

1. $\forall\, \rho \in \text{Out}^{\sigma}_{D}\colon\ \text{TS}^{T}(\rho) \leq \ell_1$,
2. $\mathbb{E}^{\sigma}_{D}(\text{TS}^{T}) \leq \ell_2$.

Theorem [BFRR14b]

The SSP-WE problem can be decided in pseudo-polynomial time and is NP-hard. Pure pseudo-polynomial-memory strategies are always sufficient and in general necessary, and satisfying strategies can be constructed in pseudo-polynomial time.

SSP-WE: pseudo-PTIME algorithm

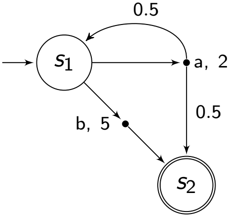


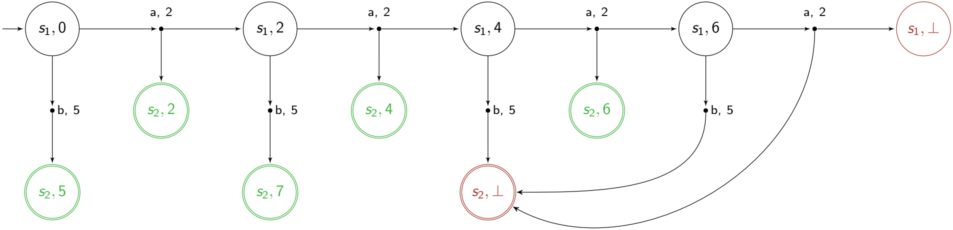
Consider SSP-WE problem for $\ell_1 = 7$ (*wc*), $\ell_2 = 4.8$ ($\mathbb{E}$).

- ▷ Reduction to the SSP-E problem on a pseudo-polynomial-size expanded MDP.

1. Build unfolding as for SSP-P problem w.r.t. worst-case threshold ℓ_1.

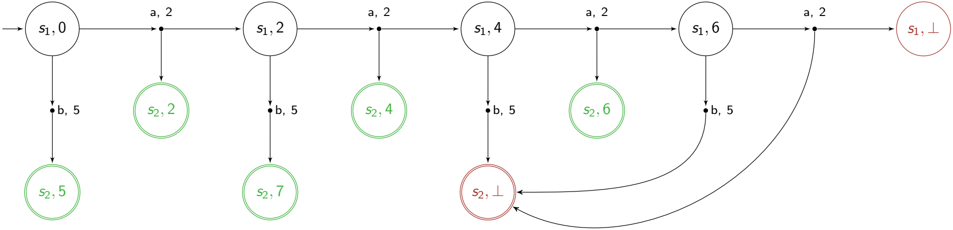
SSP-WE: pseudo-PTIME algorithm





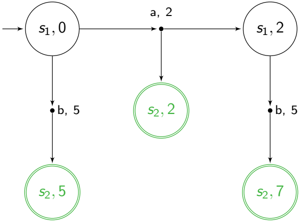
SSP-WE: pseudo-PTIME algorithm

2. Compute R, the attractor of $T' = T \times \{0, 1, \ldots, \ell_1\}$.
3. Restrict MDP to $D' = D_{\ell_1} \downharpoonright R$, the *safe* part w.r.t. SP-G.



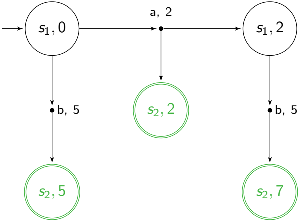
SSP-WE: pseudo-PTIME algorithm

2. Compute R, the attractor of $T' = T \times \{0, 1, \ldots, \ell_1\}$.
3. Restrict MDP to $D' = D_{\ell_1} \downharpoonright R$, the *safe* part w.r.t. SP-G.



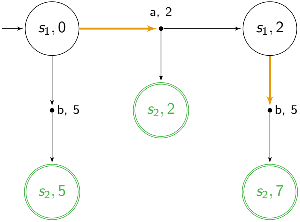
SSP-WE: pseudo-PTIME algorithm

4. Compute memoryless optimal strategy σ in D' for SSP-E.
5. Answer is YES iff $\mathbb{E}^{\sigma}_{D'}(\mathsf{TS}^{T'}) \leq \ell_2$.



SSP-WE: pseudo-PTIME algorithm

4. Compute memoryless optimal strategy σ in D' for SSP-E.
5. Answer is YES iff $\mathbb{E}^{\sigma}_{D'}(\mathsf{TS}^{T'}) \leq \ell_2$.



Here,
$\mathbb{E}^{\sigma}_{D'}(\mathsf{TS}^{T'}) = 9/2.$

SSP-WE: wrap-up

SSP	complexity	strategy
SSP-E	PTIME	pure memoryless
SSP-P	pseudo-PTIME / PSPACE-h.	pure pseudo-poly.
SSP-G	PTIME	pure memoryless
SSP-WE	pseudo-PTIME / NP-h.	pure pseudo-poly.

▷ NP-hardness ⇒ inherently harder than SSP-E and SSP-G.

Related work (non-exhaustive)

- BWC synthesis problems for mean-payoff [BFRR14b] and parity [BRR17] belong to NP $\cap$ coNP. Much more involved technically.

  $\Longrightarrow$ Additional modeling power for free w.r.t. worst-case problems.

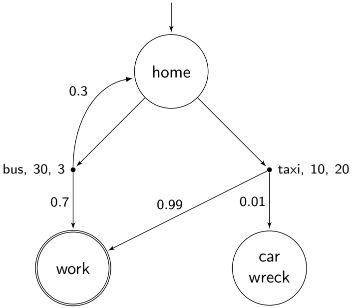
Related work (non-exhaustive)

- BWC synthesis problems for mean-payoff [BFRR14b] and parity [BRR17] belong to NP $\cap$ coNP. Much more involved technically.

  $\Longrightarrow$ Additional modeling power for free w.r.t. worst-case problems.

- Multi-dimensional extension for mean-payoff [CR15].
- Integration of BWC concepts in UPPAAL [DJL+14].
- Optimizing the expected mean-payoff under energy constraints [BKN16] or Boolean constraints [AKV16].

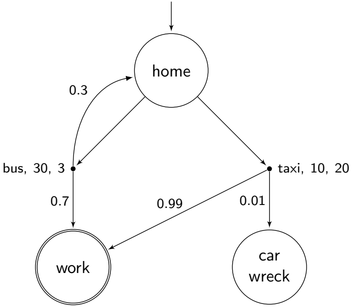
Multiple objectives $\Longrightarrow$ trade-offs



Two-dimensional weights on actions: *time* and *cost*.

Often necessary to consider trade-offs: e.g., between the probability to reach work in due time and the risks of an expensive journey.

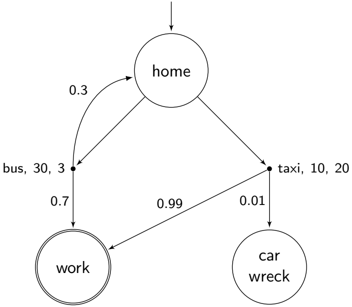
Multiple objectives $\implies$ trade-offs



SSP-P problem considers a **single percentile constraint**.

- **C1**: 80% of runs reach work in at most 40 minutes.
  - ▷ Taxi $\rightsquigarrow \leq 10$ minutes with probability $0.99 > 0.8$.

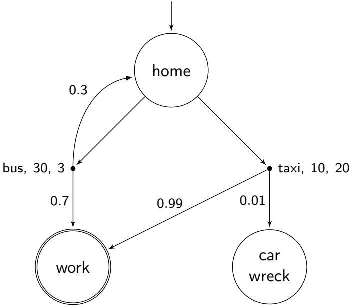
Multiple objectives $\implies$ trade-offs



SSP-P problem considers a **single percentile constraint**.

- **C1**: 80% of runs reach work in at most 40 minutes.
  - ▷ Taxi $\rightsquigarrow \leq 10$ minutes with probability $0.99 > 0.8$.
- **C2**: 50% of them cost at most 10$ to reach work.
  - ▷ Bus $\rightsquigarrow \geq 70\%$ of the runs reach work for 3$.

Multiple objectives $\implies$ trade-offs

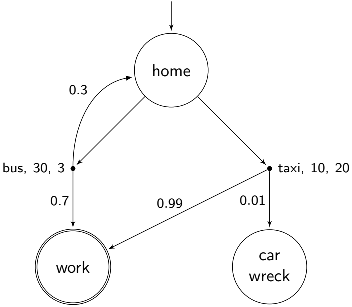


SSP-P problem considers a **single percentile constraint**.

- **C1**: 80% of runs reach work in at most 40 minutes.
  - ▷ Taxi $\rightsquigarrow \leq 10$ minutes with probability $0.99 > 0.8$.
- **C2**: 50% of them cost at most 10$ to reach work.
  - ▷ Bus $\rightsquigarrow \geq 70\%$ of the runs reach work for 3$.

Taxi $\not\models$ C2, bus $\not\models$ C1. What if we want C1 $\wedge$ C2?

Multiple objectives $\implies$ trade-offs

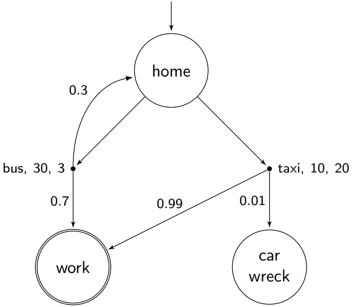


- **C1**: 80% of runs reach work in at most 40 minutes.
- **C2**: 50% of them cost at most 10$ to reach work.

Study of **multi-constraint percentile queries** [RRS15a].

- ▷ Sample strategy: bus once, then taxi. Requires *memory*.
- ▷ Another strategy: bus with probability 3/5, taxi with probability 2/5. Requires *randomness*.

Multiple objectives $\implies$ trade-offs



- **C1**: 80% of runs reach work in at most 40 minutes.
- **C2**: 50% of them cost at most 10$ to reach work.

Study of **multi-constraint percentile queries** [RRS15a].

In general, *both* memory *and* randomness are required.

$\neq$ **Previous problems.**

SSP-PQ: multi-constraint percentile queries (1/2)

SSP-PQ problem

Given d-dimensional MDP $D = (S, s_{\mathsf{init}}, A, \delta, w)$, and $q \in \mathbb{N}$ percentile constraints described by target sets $T_i \subseteq S$, dimensions $k_i \in \{1, \ldots, d\}$, value thresholds $\ell_i \in \mathbb{N}$ and probability thresholds $\alpha_i \in [0, 1] \cap \mathbb{Q}$, where $i \in \{1, \ldots, q\}$, decide if there exists a strategy σ such that query $\mathcal{Q}$ holds, with

$$\mathcal{Q} := \bigwedge_{i=1}^{q} \mathbb{P}^{\sigma}_{D}\left[\mathsf{TS}^{T_i}_{k_i} \leq \ell_i\right] \geq \alpha_i,$$

where $\mathsf{TS}^{T_i}_{k_i}$ denotes the truncated sum on dimension k_i and w.r.t. target set T_i.

Very general framework: multiple constraints related to $\neq$ dimensions, and $\neq$ target sets $\implies$ great flexibility in modeling.

SSP-PQ: multi-constraint percentile queries (2/2)

Theorem [RRS15a]

The SSP-PQ problem can be decided in

- exponential time in general,
- pseudo-polynomial time for single-dimension single-target multi-contraint queries.

It is PSPACE-hard even for single-constraint queries. Randomized exponential-memory strategies are always sufficient and in general necessary, and satisfying strategies can be constructed in exponential time.

▷ PSPACE-hardness already true for SSP-P [HK15].

$\rightsquigarrow$ SSP-PQ = wide extension for basically no price in complexity.

SSP-PQ: EXPTIME / pseudo-PTIME algorithm

1. Build an unfolded MDP D_ℓ similar to SSP-P case:
   - stop unfolding when *all* dimensions reach sum $\ell = \max_i \ell_i$.

SSP-PQ: EXPTIME / pseudo-PTIME algorithm

1. Build an unfolded MDP D_ℓ similar to SSP-P case:
   - stop unfolding when *all* dimensions reach sum $\ell = \max_i \ell_i$.

2. Maintain *single*-exponential size by defining an equivalence relation between states of D_ℓ:
   - $S_\ell \subseteq S \times (\{0, \ldots, \ell\} \cup \{\perp\})^d$,
   - pseudo-poly. if $d = 1$.

SSP-PQ: EXPTIME / pseudo-PTIME algorithm

1. Build an unfolded MDP D_ℓ similar to SSP-P case:
   - stop unfolding when *all* dimensions reach sum $\ell = \max_i \ell_i$.

2. Maintain *single*-exponential size by defining an equivalence relation between states of D_ℓ:
   - $S_\ell \subseteq S \times (\{0, \ldots, \ell\} \cup \{\bot\})^d$,
   - pseudo-poly. if $d = 1$.

3. For each constraint i, compute a target set R_i in D_ℓ:
   - $\rho \models$ constraint i in $D \iff \rho' \models \Diamond R_i$ in D_ℓ.

SSP-PQ: EXPTIME / pseudo-PTIME algorithm

1. Build an unfolded MDP D_ℓ similar to SSP-P case:
   - stop unfolding when *all* dimensions reach sum $\ell = \max_i \ell_i$.
2. Maintain *single*-exponential size by defining an equivalence relation between states of D_ℓ:
   - $S_\ell \subseteq S \times (\{0, \ldots, \ell\} \cup \{\bot\})^d$,
   - pseudo-poly. if $d = 1$.
3. For each constraint i, compute a target set R_i in D_ℓ:
   - $\rho \models$ constraint i in $D \iff \rho' \models \Diamond R_i$ in D_ℓ.
4. Solve a **multiple reachability problem** on D_ℓ.
   - Generalizes the SR problem [EKVY08, RRS15a].
   - Time polynomial in $|D_\ell|$ but exponential in q.
   - Single-dim. single target queries $\Rightarrow$ absorbing targets $\Rightarrow$ polynomial-time algorithm.

SSP-PQ: wrap-up

SSP	complexity	strategy
SSP-E	PTIME	pure memoryless
SSP-P	pseudo-PTIME / PSPACE-h.	pure pseudo-poly.
SSP-G	PTIME	pure memoryless
SSP-WE	pseudo-PTIME / NP-h.	pure pseudo-poly.
SSP-PQ	EXPTIME (p.-PTIME) / PSPACE-h.	randomized exponential

- ▷ SSP-PQ is undecidable for arbitrary weights in multi-dimensional MDPs, even with a unique target set [RRS15a].

Percentile queries: overview (1/2)

- **Wide range of payoff functions**
  - ▷ multiple reachability,
  - ▷ inf, sup, lim inf, lim sup,
  - ▷ mean-payoff ($\overline{\mathsf{MP}}$, $\underline{\mathsf{MP}}$),
  - ▷ shortest path (SP),
  - ▷ discounted sum (DS).

Percentile queries: overview (1/2)

- **Wide range of payoff functions**
  - multiple reachability,
  - inf, sup, lim inf, lim sup,
  - mean-payoff ($\overline{\mathsf{MP}}$, $\underline{\mathsf{MP}}$),
  - shortest path (SP),
  - discounted sum (DS).
- **Several variants**:
  - multi-dim. multi-constraint,
  - single-dim. multi-constraint,
  - single-constraint.

Percentile queries: overview (1/2)

- **Wide range of payoff functions**
  - multiple reachability,
  - inf, sup, lim inf, lim sup,
  - mean-payoff ($\overline{\mathsf{MP}}$, $\underline{\mathsf{MP}}$),
  - shortest path (SP),
  - discounted sum (DS).
- **Several variants**:
  - multi-dim. multi-constraint,
  - single-dim. multi-constraint,
  - single-constraint.
- For each one:
  - algorithms,
  - lower bounds,
  - memory requirements.

$\rightsquigarrow$ **Complete picture** for this new framework.

Percentile queries: overview (2/2)

	Single-constraint	Single-dim. Multi-constraint	Multi-dim. Multi-constraint
Reachability	P [Put94]	$\mathsf{P}(D)\cdot\mathsf{E}(\mathcal{Q})$ [EKVY08], PSPACE-h	—
$f \in \mathcal{F}$	P [CH09]	P	$\mathsf{P}(D)\cdot\mathsf{E}(\mathcal{Q})$ PSPACE-h.
$\overline{\mathsf{MP}}$	P [Put94]	P	P
$\underline{\mathsf{MP}}$	P [Put94]	$\mathsf{P}(D)\cdot\mathsf{E}(\mathcal{Q})$	$\mathsf{P}(D)\cdot\mathsf{E}(\mathcal{Q})$
SP	$\mathsf{P}(D)\cdot\mathsf{P}_{ps}(\mathcal{Q})$ [HK15] PSPACE-h. [HK15]	$\mathsf{P}(D)\cdot\mathsf{P}_{ps}(\mathcal{Q})$ (one target) PSPACE-h. [HK15]	$\mathsf{P}(D)\cdot\mathsf{E}(\mathcal{Q})$ PSPACE-h. [HK15]
ε-gap DS	$\mathsf{P}_{ps}(D, \mathcal{Q}, \varepsilon)$ NP-h.	$\mathsf{P}_{ps}(D, \varepsilon)\cdot\mathsf{E}(\mathcal{Q})$ NP-h.	$\mathsf{P}_{ps}(D, \varepsilon)\cdot\mathsf{E}(\mathcal{Q})$ PSPACE-h.

- $\mathcal{F} = \{\inf, \sup, \liminf, \limsup\}$
- D = model size, $\mathcal{Q}$ = query size
- $\mathsf{P}(x)$, $\mathsf{E}(x)$ and $\mathsf{P}_{ps}(x)$ resp. denote polynomial, exponential and pseudo-polynomial time in parameter x.

All results without reference are established in [RRS15a].

Percentile queries: overview (2/2)

	Single-constraint	Single-dim. Multi-constraint	Multi-dim. Multi-constraint
Reachability	P [Put94]	P(D)·E($\mathcal{Q}$) [EKVY08], PSPACE-h	—
$f \in \mathcal{F}$	P [CH09]	P	P(D)·E($\mathcal{Q}$) PSPACE-h.
$\overline{\text{MP}}$	P [Put94]	P	P
$\underline{\text{MP}}$	P [Put94]	P(D)·E($\mathcal{Q}$)	P(D)·E($\mathcal{Q}$)
SP	P(D)·P_{ps}($\mathcal{Q}$) [HK15] PSPACE-h. [HK15]	P(D)·P_{ps}($\mathcal{Q}$) (one target) PSPACE-h. [HK15]	P(D)·E($\mathcal{Q}$) PSPACE-h. [HK15]
ε-gap DS	$\text{P}_{ps}(D, \mathcal{Q}, \varepsilon)$ NP-h.	$\text{P}_{ps}(D, \varepsilon)$·E($\mathcal{Q}$) NP-h.	$\text{P}_{ps}(D, \varepsilon)$·E($\mathcal{Q}$) PSPACE-h.

In most cases, only **polynomial in the model size**.

- ▷ In practice, the query size can often be bounded while the model can be very large.

Percentile queries: overview (2/2)

	Single-constraint	Single-dim. Multi-constraint	Multi-dim. Multi-constraint
Reachability	P [Put94]	P(D)·E($\mathcal{Q}$) [EKVY08], PSPACE-h	—
$f \in \mathcal{F}$	P [CH09]	P	P(D)·E($\mathcal{Q}$) PSPACE-h.
$\overline{\mathsf{MP}}$	P [Put94]	P	P
$\underline{\mathsf{MP}}$	P [Put94]	P(D)·E($\mathcal{Q}$)	P(D)·E($\mathcal{Q}$)
SP	P(D)·P_{ps}($\mathcal{Q}$) [HK15] PSPACE-h. [HK15]	P(D)·P_{ps}($\mathcal{Q}$) (one target) PSPACE-h. [HK15]	P(D)·E($\mathcal{Q}$) PSPACE-h. [HK15]
ε-gap DS	$\mathsf{P}_{ps}(D, \mathcal{Q}, \varepsilon)$ NP-h.	$\mathsf{P}_{ps}(D, \varepsilon)$·E($\mathcal{Q}$) NP-h.	$\mathsf{P}_{ps}(D, \varepsilon)$·E($\mathcal{Q}$) PSPACE-h.

Four groups of results.

1. **Reachability**. Algorithm based on multi-objective linear programming (LP) in [EKVY08]. We refine the complexity analysis, provide LBs and tractable subclasses.
   - ▷ **Useful tool** for many payoff functions!

Percentile queries: overview (2/2)

	Single-constraint	Single-dim. Multi-constraint	Multi-dim. Multi-constraint
Reachability	P [Put94]	P(D)·E($\mathcal{Q}$) [EKVY08], PSPACE-h	—
$f \in \mathcal{F}$	P [CH09]	P	P(D)·E($\mathcal{Q}$) PSPACE-h.
$\overline{\text{MP}}$	P [Put94]	P	P
$\underline{\text{MP}}$	P [Put94]	P(D)·E($\mathcal{Q}$)	P(D)·E($\mathcal{Q}$)
SP	P(D)·P_{ps}($\mathcal{Q}$) [HK15] PSPACE-h. [HK15]	P(D)·P_{ps}($\mathcal{Q}$) (one target) PSPACE-h. [HK15]	P(D)·E($\mathcal{Q}$) PSPACE-h. [HK15]
ε-gap DS	$\text{P}_{ps}(D, \mathcal{Q}, \varepsilon)$ NP-h.	$\text{P}_{ps}(D, \varepsilon)$·E($\mathcal{Q}$) NP-h.	$\text{P}_{ps}(D, \varepsilon)$·E($\mathcal{Q}$) PSPACE-h.

Four groups of results.

2 $\mathcal{F}$ **and** $\overline{\text{MP}}$. Easiest cases.

- inf and sup: reduction to *multiple reachability*.
- lim inf, lim sup and $\overline{\text{MP}}$: *maximal end-component* (MEC) decomposition + reduction to multiple reachability.

Percentile queries: overview (2/2)

	Single-constraint	Single-dim. Multi-constraint	Multi-dim. Multi-constraint
Reachability	P [Put94]	P(D)·E($\mathcal{Q}$) [EKVY08], PSPACE-h	—
$f \in \mathcal{F}$	P [CH09]	P	P(D)·E($\mathcal{Q}$) PSPACE-h.
$\overline{\mathsf{MP}}$	P [Put94]	P	P
$\underline{\mathsf{MP}}$	P [Put94]	P(D)·E($\mathcal{Q}$)	P(D)·E($\mathcal{Q}$)
SP	P(D)·P_{ps}($\mathcal{Q}$) [HK15] PSPACE-h. [HK15]	P(D)·P_{ps}($\mathcal{Q}$) (one target) PSPACE-h. [HK15]	P(D)·E($\mathcal{Q}$) PSPACE-h. [HK15]
ε-gap DS	$\mathsf{P}_{ps}(D, \mathcal{Q}, \varepsilon)$ NP-h.	$\mathsf{P}_{ps}(D, \varepsilon)$·E($\mathcal{Q}$) NP-h.	$\mathsf{P}_{ps}(D, \varepsilon)$·E($\mathcal{Q}$) PSPACE-h.

Four groups of results.

3. $\underline{\mathsf{MP}}$. Technically involved.
   - ▷ Inside MECs: (a) strategies satisfying *maximal subsets of constraints*, (b) combine them linearly.
   - ▷ Overall: write an LP combining multiple reachability toward MECs and those linear combinations equations.

Percentile queries: overview (2/2)

	Single-constraint	Single-dim. Multi-constraint	Multi-dim. Multi-constraint
Reachability	P [Put94]	P(D)·E($\mathcal{Q}$) [EKVY08], PSPACE-h	—
$f \in \mathcal{F}$	P [CH09]	P	P(D)·E($\mathcal{Q}$) PSPACE-h.
$\overline{\text{MP}}$	P [Put94]	P	P
$\underline{\text{MP}}$	P [Put94]	P(D)·E($\mathcal{Q}$)	P(D)·E($\mathcal{Q}$)
SP	P(D)·P$_{ps}$($\mathcal{Q}$) [HK15] PSPACE-h. [HK15]	P(D)·P$_{ps}$($\mathcal{Q}$) (one target) PSPACE-h. [HK15]	P(D)·E($\mathcal{Q}$) PSPACE-h. [HK15]
ε-gap DS	P$_{ps}$($D, \mathcal{Q}, \varepsilon$) NP-h.	P$_{ps}$(D, ε)·E($\mathcal{Q}$) NP-h.	P$_{ps}$(D, ε)·E($\mathcal{Q}$) PSPACE-h.

Four groups of results.

4 **SP and DS**. Based on *unfoldings* and multiple reachability.
- ▷ For SP, we bound the size of the unfolding by *node merging*.
- ▷ For DS, we can only *approximate* the answer in general. Need to analyze the cumulative error due to necessary *roundings*.

Summary: stochastic shortest path problem

- **SSP-E:** minimize the expected sum to target.
  - ▷ Actual outcomes may vary greatly.

Summary: stochastic shortest path problem

- **SSP-E:** minimize the expected sum to target.
  - ▷ Actual outcomes may vary greatly.
- **SSP-P:** maximize the probability of acceptable performance.
  - ▷ No control over the quality of bad runs, no average-case performance.

Summary: stochastic shortest path problem

- **SSP-E:** minimize the expected sum to target.
  - ▷ Actual outcomes may vary greatly.
- **SSP-P:** maximize the probability of acceptable performance.
  - ▷ No control over the quality of bad runs, no average-case performance.
- **SP-G:** maximize the worst-case performance, extreme risk-aversion.
  - ▷ Strict worst-case guarantees, no average-case performance.

Summary: stochastic shortest path problem

- **SSP-E:** minimize the expected sum to target.
  - ▷ Actual outcomes may vary greatly.
- **SSP-P:** maximize the probability of acceptable performance.
  - ▷ No control over the quality of bad runs, no average-case performance.
- **SP-G:** maximize the worst-case performance, extreme risk-aversion.
  - ▷ Strict worst-case guarantees, no average-case performance.
- **SSP-WE:** SSP-E $\cap$ SP-G.
  - ▷ Based on beyond worst-case synthesis [BFRR14b, BFRR14a].

Summary: stochastic shortest path problem

- **SSP-E:** minimize the expected sum to target.
  - ▷ Actual outcomes may vary greatly.
- **SSP-P:** maximize the probability of acceptable performance.
  - ▷ No control over the quality of bad runs, no average-case performance.
- **SP-G:** maximize the worst-case performance, extreme risk-aversion.
  - ▷ Strict worst-case guarantees, no average-case performance.
- **SSP-WE:** SSP-E $\cap$ SP-G.
  - ▷ Based on beyond worst-case synthesis [BFRR14b, BFRR14a].
- **SSP-PQ:** extends SSP-P to multi-constraint percentile queries [RRS15a].
  - ▷ Multi-dimensional, flexible, trade-offs.
  - ▷ Complexity usually acceptable w.r.t. model size.

Thank you! Any question?

References I

Shaull Almagor, Orna Kupferman, and Yaron Velner.
Minimizing expected cost under hard boolean constraints, with applications to quantitative synthesis.
In Josée Desharnais and Radha Jagadeesan, editors, 27th International Conference on Concurrency Theory, CONCUR 2016, August 23-26, 2016, Québec City, Canada, volume 59 of LIPIcs, pages 9:1–9:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

Tomás Brázdil, Taolue Chen, Vojtech Forejt, Petr Novotný, and Aistis Simaitis.
Solvency markov decision processes with interest.
In Anil Seth and Nisheeth K. Vishnoi, editors, IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2013, December 12-14, 2013, Guwahati, India, volume 24 of LIPIcs, pages 487–499. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

Romain Brenguier, Lorenzo Clemente, Paul Hunter, Guillermo A. Pérez, Mickael Randour, Jean-François Raskin, Ocan Sankur, and Mathieu Sassolas.
Non-zero sum games for reactive synthesis.
In Adrian-Horia Dediu, Jan Janousek, Carlos Martín-Vide, and Bianca Truthe, editors, Language and Automata Theory and Applications - 10th International Conference, LATA 2016, Prague, Czech Republic, March 14-18, 2016, Proceedings, volume 9618 of Lecture Notes in Computer Science, pages 3–23. Springer, 2016.

Véronique Bruyère, Emmanuel Filiot, Mickael Randour, and Jean-François Raskin.
Expectations or guarantees? I want it all! A crossroad between games and MDPs.
In Fabio Mogavero, Aniello Murano, and Moshe Y. Vardi, editors, Proceedings 2nd International Workshop on Strategic Reasoning, SR 2014, Grenoble, France, April 5-6, 2014, volume 146 of EPTCS, pages 1–8, 2014.

References II

Véronique Bruyère, Emmanuel Filiot, Mickael Randour, and Jean-François Raskin.
Meet your expectations with guarantees: Beyond worst-case synthesis in quantitative games.
In Ernst W. Mayr and Natacha Portier, editors, 31st International Symposium on Theoretical Aspects of Computer Science, STACS 2014, March 5-8, 2014, Lyon, France, volume 25 of LIPIcs, pages 199–213. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2014.

Thomas Brihaye, Gilles Geeraerts, Axel Haddad, and Benjamin Monmege.
Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games.
Acta Inf., 54(1):85–125, 2017.

Udi Boker and Thomas A. Henzinger.
Exact and approximate determinization of discounted-sum automata.
Logical Methods in Computer Science, 10(1), 2014.

Udi Boker, Thomas A. Henzinger, and Jan Otop.
The target discounted-sum problem.
In 30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015, pages 750–761. IEEE Computer Society, 2015.

Tomás Brázdil, Antonín Kucera, and Petr Novotný.
Optimizing the expected mean payoff in energy Markov decision processes.
In Cyrille Artho, Axel Legay, and Doron Peled, editors, Automated Technology for Verification and Analysis - 14th International Symposium, ATVA 2016, Chiba, Japan, October 17-20, 2016, Proceedings, volume 9938 of Lecture Notes in Computer Science, pages 32–49, 2016.

Raphaël Berthon, Mickael Randour, and Jean-François Raskin.
Threshold constraints with guarantees for parity objectives in markov decision processes.
CoRR, abs/1702.05472, 2017.

References III

Dimitri P. Bertsekas and John N. Tsitsiklis.
An analysis of stochastic shortest path problems.
Mathematics of Operations Research, 16(3):580–595, 1991.

Krishnendu Chatterjee, Laurent Doyen, Mickael Randour, and Jean-François Raskin.
Looking at mean-payoff and total-payoff through windows.
Inf. Comput., 242:25–52, 2015.

Krishnendu Chatterjee, Vojtech Forejt, and Dominik Wojtczak.
Multi-objective discounted reward verification in graphs and mdps.
In Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov, editors, Logic for Programming, Artificial Intelligence, and Reasoning - 19th International Conference, LPAR-19, Stellenbosch, South Africa, December 14-19, 2013. Proceedings, volume 8312 of Lecture Notes in Computer Science, pages 228–242. Springer, 2013.

Boris V. Cherkassky, Andrew V. Goldberg, and Tomasz Radzik.
Shortest paths algorithms: Theory and experimental evaluation.
Math. programming, 73(2):129–174, 1996.

Krishnendu Chatterjee and Thomas A. Henzinger.
Probabilistic systems with limsup and liminf objectives.
In Margaret Archibald, Vasco Brattka, Valentin Goranko, and Benedikt Löwe, editors, Infinity in Logic and Computation, volume 5489 of Lecture Notes in Computer Science, pages 32–45. Springer Berlin Heidelberg, 2009.

References IV

Lorenzo Clemente and Jean-François Raskin.
Multidimensional beyond worst-case and almost-sure problems for mean-payoff objectives.
In 30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015, pages 257–268. IEEE Computer Society, 2015.

Luca de Alfaro.
Computing minimum and maximum reachability times in probabilistic systems.
In Jos C. M. Baeten and Sjouke Mauw, editors, CONCUR '99: Concurrency Theory, 10th International Conference, Eindhoven, The Netherlands, August 24-27, 1999, Proceedings, volume 1664 of Lecture Notes in Computer Science, pages 66–81. Springer, 1999.

Alexandre David, Peter Gjøl Jensen, Kim Guldstrand Larsen, Axel Legay, Didier Lime, Mathias Grund Sørensen, and Jakob Haahr Taankvist.
On time with minimal expected cost!
In Franck Cassez and Jean-François Raskin, editors, Automated Technology for Verification and Analysis - 12th International Symposium, ATVA 2014, Sydney, NSW, Australia, November 3-7, 2014, Proceedings, volume 8837 of Lecture Notes in Computer Science, pages 129–145. Springer, 2014.

Kousha Etessami, Marta Z. Kwiatkowska, Moshe Y. Vardi, and Mihalis Yannakakis.
Multi-objective model checking of markov decision processes.
Logical Methods in Computer Science, 4(4), 2008.

Emmanuel Filiot, Raffaella Gentilini, and Jean-François Raskin.
Quantitative languages defined by functional automata.
Logical Methods in Computer Science, 11(3), 2015.

References V

Vojtech Forejt, Marta Z. Kwiatkowska, Gethin Norman, David Parker, and Hongyang Qu.
Quantitative multi-objective verification for probabilistic systems.
In Parosh Aziz Abdulla and K. Rustan M. Leino, editors, Tools and Algorithms for the Construction and Analysis of Systems - 17th International Conference, TACAS 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011, Saarbrücken, Germany, March 26-April 3, 2011. Proceedings, volume 6605 of Lecture Notes in Computer Science, pages 112–127. Springer, 2011.

Oded Goldreich.
On promise problems: A survey.
In Oded Goldreich, Arnold L. Rosenberg, and Alan L. Selman, editors, Theoretical Computer Science, Essays in Memory of Shimon Even, volume 3895 of Lecture Notes in Computer Science, pages 254–290. Springer, 2006.

Christoph Haase and Stefan Kiefer.
The odds of staying on budget.
In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II, volume 9135 of Lecture Notes in Computer Science, pages 234–246. Springer, 2015.

Christoph Haase and Stefan Kiefer.
The complexity of the Kth largest subset problem and related problems.
Inf. Process. Lett., 116(2):111–115, 2016.

Serge Haddad and Benjamin Monmege.
Reachability in MDPs: Refining convergence of value iteration.
In Joël Ouaknine, Igor Potapov, and James Worrell, editors, Reachability Problems - 8th International Workshop, RP 2014, Oxford, UK, September 22-24, 2014. Proceedings, volume 8762 of Lecture Notes in Computer Science, pages 125–137. Springer, 2014.

References VI

Leonid Khachiyan, Endre Boros, Konrad Borys, Khaled M. Elbassioni, Vladimir Gurvich, Gábor Rudolf, and Jihui Zhao.
On short paths interdiction problems: Total and node-wise limited interdiction.
Theory Comput. Syst., 43(2):204–233, 2008.

Yoshio Ohtsubo.
Optimal threshold probability in undiscounted Markov decision processes with a target set.
Applied Math. and Computation, 149(2):519 – 532, 2004.

Martin L. Puterman.
Markov Decision Processes: Discrete Stochastic Dynamic Programming.
John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

Mickael Randour.
Reconciling rationality and stochasticity: Rich behavioral models in two-player games.
CoRR, abs/1603.05072, 2016.
GAMES 2016, the 5th World Congress of the Game Theory Society, Maastricht, Netherlands.

Mickael Randour, Jean-François Raskin, and Ocan Sankur.
Percentile queries in multi-dimensional Markov decision processes.
In Daniel Kroening and Corina S. Pasareanu, editors, Computer Aided Verification - 27th International Conference, CAV 2015, San Francisco, CA, USA, July 18-24, 2015, Proceedings, Part I, volume 9206 of Lecture Notes in Computer Science, pages 123–139. Springer, 2015.

References VII

Mickael Randour, Jean-François Raskin, and Ocan Sankur.
Variations on the stochastic shortest path problem.
In Deepak D'Souza, Akash Lal, and Kim Guldstrand Larsen, editors, Verification, Model Checking, and Abstract Interpretation - 16th International Conference, VMCAI 2015, Mumbai, India, January 12-14, 2015. Proceedings, volume 8931 of Lecture Notes in Computer Science, pages 1–18. Springer, 2015.

Masahiko Sakaguchi and Yoshio Ohtsubo.
Markov decision processes associated with two threshold probability criteria.
Journal of Control Theory and Applications, 11(4):548–557, 2013.

Stephen D. Travers.
The complexity of membership problems for circuits over sets of integers.
Theor. Comput. Sci., 369(1-3):211–229, 2006.

Michael Ummels and Christel Baier.
Computing quantiles in markov reward models.
In Frank Pfenning, editor, Foundations of Software Science and Computation Structures - 16th International Conference, FOSSACS 2013, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2013, Rome, Italy, March 16-24, 2013. Proceedings, volume 7794 of Lecture Notes in Computer Science, pages 353–368. Springer, 2013.

Multi-constraint queries for DS

Multi-constraint percentile problem for DS

Given d-dimensional MDP $D = (S, s_{\mathsf{init}}, A, \delta, w)$, and $q \in \mathbb{N}$ percentile constraints described by discount factors $\lambda_i \in]0, 1[\cap \mathbb{Q}$, dimensions $l_i \in \{1, \ldots, d\}$, value thresholds $v_i \in \mathbb{Q}$ and probability thresholds $\alpha_i \in [0, 1] \cap \mathbb{Q}$, where $i \in \{1, \ldots, q\}$, decide if there exists a strategy σ such that query $\mathcal{Q}$ holds, with

$$\mathcal{Q} := \bigwedge_{i=1}^{q} \mathbb{P}^{\sigma}_{M, s_{\mathsf{init}}} \left[\mathsf{DS}^{\lambda_i}_{l_i} \geq v_i \right] \geq \alpha_i,$$

where $\mathsf{DS}^{\lambda_i}_{l_i}(\rho) = \sum_{j=1}^{\infty} \lambda_i^j \cdot w_{l_i}(a_j)$ denotes the discounted sum on dimension l_i and w.r.t. discount factor λ_i.

We allow **arbitrary** weights for this payoff.

Precise discounted sum problem is hard

Precise DS problem

Given value $t \in \mathbb{Q}$, and discount factor $\lambda \in]0,1[$, does there exist an infinite binary sequence $\tau = \tau_1\tau_2\tau_3\ldots \in \{0,1\}^\omega$ such that $\sum_{j=1}^{\infty} \lambda^j \cdot \tau_j = t$?

- Reduces to an almost-sure percentile problem on a single-state 2-dim. MDP.
- **Still not known to be decidable!**
  - $\rightsquigarrow$ Related to open questions such as the *universality problem for discounted-sum automata* [BHO15, CFW13, BH14].

Precise discounted sum problem is hard

Precise DS problem

Given value $t \in \mathbb{Q}$, and discount factor $\lambda \in]0, 1[$, does there exist an infinite binary sequence $\tau = \tau_1\tau_2\tau_3 \ldots \in \{0, 1\}^\omega$ such that $\sum_{j=1}^{\infty} \lambda^j \cdot \tau_j = t$?

- ▷ Reduces to an almost-sure percentile problem on a single-state 2-dim. MDP.
- ▷ **Still not known to be decidable!**
  - ⇝ Related to open questions such as the *universality problem for discounted-sum automata* [BHO15, CFW13, BH14].

We cannot solve the exact problem but we can approximate correct answers.

ε-gap percentile problem (1/3)

- Classical decision problem.
  - Two types of inputs: *yes*-inputs and *no*-inputs.
  - Correct answers required for both types.

ε-gap percentile problem (1/3)

- Classical decision problem.
  - Two types of inputs: *yes*-inputs and *no*-inputs.
  - Correct answers required for both types.
- Promise problem [Gol06].
  - Three types: *yes*-inputs, *no*-inputs, *remaining* inputs.
  - Correct answers required for yes-inputs and no-inputs, arbitrary answer OK for the remaining ones.

ε-gap percentile problem (1/3)

- Classical decision problem.
  - Two types of inputs: *yes*-inputs and *no*-inputs.
  - Correct answers required for both types.
- Promise problem [Gol06].
  - Three types: *yes*-inputs, *no*-inputs, *remaining* inputs.
  - Correct answers required for yes-inputs and no-inputs, arbitrary answer OK for the remaining ones.
- ε-gap problem.
  - The uncertainty zone can be made **arbitrarily small**, parametrized by value $\varepsilon > 0$.

ε-gap percentile problem (2/3)

We build an **algorithm**.

- Inputs: query $\mathcal{Q}$ and precision factor $\varepsilon > 0$.
- Output: Yes, No or Unknown.
  - ▷ If Yes, then a strategy exists and can be synthesized.
  - ▷ If No, then no strategy exists.
  - ▷ Answer Unknown can only be output within an uncertainty zone of size $\sim \varepsilon$.
    - $\Rightarrow$ **Incremental approximation scheme.**

ε-gap percentile problem (3/3)

Theorem

There is an algorithm that, given an MDP, a percentile query $\mathcal{Q}$ for the DS and a precision factor $\varepsilon > 0$, solves the following ε-gap problem in exponential time. It answers

- Yes if **there is** a strategy satisfying query $\mathcal{Q}_{2\cdot\varepsilon}$;
- No if **there is no** strategy satisfying query $\mathcal{Q}_{-2\cdot\varepsilon}$;
- and arbitrarily otherwise.

▷ **Shifted query**: $\mathcal{Q}_x \equiv \mathcal{Q}$ with value thresholds $v_i + x$ (all other things being equal).

ε-gap percentile problem (3/3)

Theorem

There is an algorithm that, given an MDP, a percentile query $\mathcal{Q}$ for the DS and a precision factor $\varepsilon > 0$, solves the following ε-gap problem in exponential time. It answers

- Yes if **there is** a strategy satisfying query $\mathcal{Q}_{2\cdot\varepsilon}$;
- No if **there is no** strategy satisfying query $\mathcal{Q}_{-2\cdot\varepsilon}$;
- and arbitrarily otherwise.

▷ **Shifted query**: $\mathcal{Q}_x \equiv \mathcal{Q}$ with value thresholds $v_i + x$ (all other things being equal).

\+ PSPACE-hard ($d \geq 2$, subset-sum games [Tra06]), NP-hard for $q = 1$ (K-th largest subset problem [HK16]), exponential memory sufficient and necessary.

Algorithm: key ideas

1. Goal: multiple reachability over appropriate *unfolding*.

Algorithm: key ideas

1. Goal: multiple reachability over appropriate *unfolding*.
2. **Finite unfolding?**
   - ▷ Sums not necessarily increasing ($\neq$ SP).
     - $\Rightarrow$ Not easy to know when to stop.

Algorithm: key ideas

1. Goal: multiple reachability over appropriate *unfolding*.
2. **Finite unfolding?**
   - ▷ Sums not necessarily increasing ($\neq$ SP).
     - $\Rightarrow$ Not easy to know when to stop.
   - ▷ Use the **discount factor**.
     - $\Rightarrow$ Weights contribute less and less to the sum along a run.
     - $\Rightarrow$ The range of possible futures narrows the deeper we go.
     - $\Rightarrow$ Cutting all branches after a pseudo-polynomial depth changes the overall sum by at most $\varepsilon/2$.

Algorithm: key ideas

1. Goal: multiple reachability over appropriate *unfolding*.
2. Pseudo-polynomial depth.
   - ▷ **2-exponential** unfolding overall!

Algorithm: key ideas

1. Goal: multiple reachability over appropriate *unfolding*.
2. Pseudo-polynomial depth.
   - ▷ **2-exponential** unfolding overall!
3. **Reduce the overall size?**
   - ▷ No direct merging of nodes (no integer labels, $\neq$ SP), too many possible label values.
   - ▷ Introduce a **rounding** scheme of the numbers involved (inspired by [BCF+13]).
     - $\Rightarrow$ We bound the error due to cumulated roundings by $\varepsilon/2$.
     - $\Rightarrow$ Single-exponential width.

Algorithm: key ideas

1. Goal: multiple reachability over appropriate *unfolding*.
2. Pseudo-polynomial depth.
3. Single-exponential width.
4. **Leaf labels are off by at most ε.** Classify each leaf w.r.t. each constraint.
   - $\sim$ Same idea as for SP.
     - $\Rightarrow$ Defining target sets for multiple reachability.
   - $\triangleright$ Leaves can be good, bad or uncertain (if too close to threshold).

Algorithm: key ideas

1. Goal: multiple reachability over appropriate *unfolding*.
2. Pseudo-polynomial depth.
3. Single-exponential width.
4. **Leaf labels are off by at most ε.** Classify each leaf w.r.t. each constraint.
   - Leaves can be good, bad or uncertain (if too close to threshold).
5. Finally, **two multiple reachability problems** to solve.
   - If OK for good leaves, then answer Yes.
   - If KO for good but OK for uncertain, then answer Unknown.
   - If KO for both, then answer No.

Algorithm: key ideas

1. Goal: multiple reachability over appropriate *unfolding*.
2. Pseudo-polynomial depth.
3. Single-exponential width.
4. **Leaf labels are off by at most ε.** Classify each leaf w.r.t. each constraint.
   - Leaves can be good, bad or uncertain (if too close to threshold).
5. Finally, **two multiple reachability problems** to solve.
   - If OK for good leaves, then answer Yes.
   - If KO for good but OK for uncertain, then answer Unknown.
   - If KO for both, then answer No.

That solves the ε-gap problem.