Percentile Queries in Multi-Dimensional Markov Decision Processes

Mickael Randour (LSV - CNRS & ENS Cachan)
Jean-François Raskin (ULB)  Ocan Sankur (ULB)

05.02.2015

Dagstuhl seminar “Non-zero-sum games and control”
The talk in one slide

Strategy synthesis for Markov Decision Processes (MDPs)

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

Multi-constraint percentile queries: generalizes the problem to multiple dimensions, multiple constraints.
Advertisement

Full paper available on arXiv [RRS14]: abs/1410.4801
1. Context, MDPs, Strategies
2. Percentile Queries
3. Shortest Path
4. Discounted Sum
5. Conclusion
1. Context, MDPs, Strategies

2. Percentile Queries

3. Shortest Path

4. Discounted Sum

5. Conclusion
Verification and synthesis:

- a reactive **system** to *control*,
- an *interacting environment*,
- a **specification** to *enforce*. 
Context

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.
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.
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

- system description
- environment description
- informal specification

model as an MDP
model as a winning objective

synthesis

is there a winning strategy?

- no
  - empower system capabilities or weaken specification requirements
- yes
  - strategy = controller
Strategy (policy) synthesis for MDPs

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

- System description
- Environment description
- Informal specification

Model as an MDP
Model as a winning objective

Synthesis

Is there a winning strategy?

- No: Empower system capabilities or weaken specification requirements
- Yes: Strategy = controller
Strategy (policy) synthesis for MDPs

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

1. 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** $M = (S, A, \delta, w)$
  - finite sets of states $S$ and actions $A$
  - probabilistic transition $\delta: S \times A \rightarrow \mathcal{D}(S)$
  - weight function $w: A \rightarrow \mathbb{Z}^d$

- **Run** (or play): $\rho = s_1a_1 \ldots a_{n-1}s_n \ldots$ such that $\delta(s_i, a_i, s_{i+1}) > 0$ for all $i \geq 1$
  - set of runs $\mathcal{R}(M)$
  - set of histories (finite runs) $\mathcal{H}(M)$

- **Strategy** $\sigma: \mathcal{H}(M) \rightarrow \mathcal{D}(A)$
  - $\forall h$ ending in $s$, $\text{Supp}(\sigma(h)) \in A(s)$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

Sample run $\rho = s_1$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

Sample run $\rho = s_1 a_1$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

Sample run $\rho = s_1 a_1 s_2$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

Sample run $\rho = s_1a_1s_2a_2$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

Sample run $\rho = s_1 a_1 s_2 a_2 s_1$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

Sample run $\rho = s_1a_1s_2a_2s_1a_1$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

Sample run $\rho = s_1 a_1 s_2 a_2 s_1 a_1 s_2$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

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 $\sigma$

Sample run $\rho = s_1a_1s_2a_2s_1a_1s_2a_2s_3$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

Sample run $\rho = s_1a_1s_2a_2s_1a_1s_2a_2s_3a_3$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

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 $\sigma$

Sample run $\rho = s_1a_1s_2a_2s_1a_1s_2a_2s_3a_3s_4a_4$
Markov decision processes

Sample *pure memoryless* strategy $\sigma$

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 $\sigma$

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 $\sigma$

Sample run $\rho = s_1a_1s_2a_2s_1a_1s_2a_2(s_3a_3s_4a_4)\omega$

Other possible run $\rho' = s_1a_1s_2a_2(s_3a_3s_4a_4)\omega$

- Strategies may use
  - finite or infinite *memory*
  - *randomness*

- **Payoff functions** map runs to numerical values
  - truncated sum up to $T = \{s_3\}$:
    $$TS^T(\rho) = 2, \quad TS^T(\rho') = 1$$
  - mean-payoff: $\text{MP}(\rho) = \text{MP}(\rho') = 1/2$
  - many more
Markov chains

Once initial state $s_{\text{init}}$ and strategy $\sigma$ fixed, fully stochastic process

$\leadsto$ Markov chain (MC)
Markov chains

Once initial state $s_{\text{init}}$ and strategy $\sigma$ fixed, fully stochastic process

$\sim$ Markov chain (MC)

State space $=$ product of the MDP and the memory of $\sigma$
Markov chains

Once initial state $s_{\text{init}}$ and strategy $\sigma$ fixed, fully stochastic process

$\sim$ Markov chain \((MC)\) 

State space $=$ product of the MDP and the memory of $\sigma$

\begin{itemize}
    \item Event $\mathcal{E} \subseteq \mathcal{R}(M)$
    \begin{itemize}
        \item probability $P_{M, s_{\text{init}}}^{\sigma}(\mathcal{E})$
    \end{itemize}
    \item Measurable $f : \mathcal{R}(M) \rightarrow (\mathbb{R} \cup \{-\infty, \infty\})^d$
    \begin{itemize}
        \item expected value $E_{M, s_{\text{init}}}^{\sigma}(f)$
    \end{itemize}
\end{itemize}
1. Context, MDPs, Strategies

2. Percentile Queries

3. Shortest Path

4. Discounted Sum

5. Conclusion
Single-constraint percentile problem

Ensuring a given performance level with sufficient probability

- uni-dimensional weight function $w : A \rightarrow \mathbb{Z}$ and payoff function $f : \mathcal{R}(M) \rightarrow \mathbb{R} \cup \{-\infty, \infty\}$
- well-studied for various payoffs

Given MDP $M = (\mathcal{S}, \mathcal{A}, \delta, w)$, initial state $s_{init}$, payoff function $f$, value threshold $v \in \mathbb{Q}$, and probability threshold $\alpha \in [0, 1] \cap \mathbb{Q}$, decide if there exists a strategy $\sigma$ such that

$$\mathbb{P}_{M, s_{init}}^\sigma \left[ \{ \rho \in \mathcal{R}_{s_{init}}(M) \mid f(\rho) \geq v \} \right] \geq \alpha.$$
Single-constraint percentile problem

Ensuring a given performance level with sufficient probability

- uni-dimensional weight function \( w : A \rightarrow \mathbb{Z} \) and payoff function \( f : \mathcal{R}(M) \rightarrow \mathbb{R} \cup \{-\infty, \infty\} \)
- well-studied for various payoffs

---

Single-constraint percentile problem

Given MDP \( M = (S, A, \delta, w) \), initial state \( s_{\text{init}} \), payoff function \( f \), value threshold \( v \in \mathbb{Q} \), and probability threshold \( \alpha \in [0, 1] \cap \mathbb{Q} \), decide if there exists a strategy \( \sigma \) such that

\[
P_{M, s_{\text{init}}}^{\sigma} \left[ \{ \rho \in \mathcal{R}_{s_{\text{init}}}(M) \mid f(\rho) \geq v \} \right] \geq \alpha.
\]

- percentile constraint, often \( P_{M, s_{\text{init}}}^{\sigma} [f \geq v] \geq \alpha \)
Illustration: stochastic shortest path problem

Shortest path (SP) 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]
Illustration: stochastic shortest path problem

Shortest path (SP) 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]

For SP, we focus on MDPs with *positive weights*

▷ **Truncated sum** payoff function for \( \rho = s_1a_1s_2a_2 \ldots \) and target set \( T \):

\[
TS^T(\rho) = \begin{cases} 
\sum_{j=1}^{n-1} w(a_j) & \text{if } s_n \text{ first visit of } T \\
\infty & \text{if } T \text{ is never reached}
\end{cases}
\]
Illustration: stochastic shortest path problem

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.
Illustration: stochastic shortest path problem

Classical problem considers only a single percentile constraint.

- **C1**: 80% of runs reach work in at most 40 minutes.
  - Taxi \( \sim \leq 10 \) minutes with probability 0.99 > 0.8.
Illustration: stochastic shortest path problem

Classical problem considers only a **single percentile constraint**.

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

- **C2**: 50% of them cost at most 10$ to reach work.
  - Bus $\sim \geq 70\%$ of the runs reach work for 3$. 
Illustration: stochastic shortest path problem

Classical problem considers only a single percentile constraint.

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

Taxi $\not\models C2$, bus $\not\models C1$. What if we want $C1 \land C2$?
Illustration: stochastic shortest path problem

- **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**.

- Sample strategy: bus once, then taxi. Requires *memory*.
- Another strategy: bus with probability 3/5, taxi with probability 2/5. Requires *randomness*.
Illustration: stochastic shortest path problem

- **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**.

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

≠ classical problems (single constraint, expected value, etc)
Multi-constraint percentile problem

Given a $d$-dimensional MDP $M = (S, A, \delta, w)$, initial state $s_{\text{init}}$, payoff function $f$, and $q \in \mathbb{N}$ percentile constraints described by 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 $\sigma$ such that query $Q$ holds, with

$$Q := \bigwedge_{i=1}^{q} \mathbb{P}_{M, s_{\text{init}}}^{\sigma} [f_{l_i} \geq v_i] \geq \alpha_i.$$ 

**Very general framework** allowing for: multiple constraints related to $\neq$ or $=$ dimensions, $\neq$ value and probability thresholds.

→ For SP, even $\neq$ targets for each constraint.

→ Great flexibility in modeling applications.
Results overview (1/2)

- **Wide range of payoff functions**
  - multiple reachability,
  - mean-payoff (\(\overline{MP}, MP\)),
  - discounted sum (DS).
  - inf, sup, lim inf, lim sup,
  - shortest path (SP),
Results overview (1/2)

- **Wide range of payoff functions**
  - multiple reachability,
  - mean-payoff (\(\overline{\text{MP}}, \text{MP}\)),
  - discounted sum (DS).

- **Several variants:**
  - multi-dim. multi-constraint,
  - single-constraint.

  - inf, sup, lim inf, lim sup,
  - shortest path (SP),
  - single-dim. multi-constraint,
Results overview (1/2)

- **Wide range of payoff functions**
  - multiple reachability,
  - mean-payoff ($\overline{MP}$, $MP$),
  - discounted sum (DS).

- **Several variants**:
  - multi-dim. multi-constraint,
  - single-constraint.

- For each one:
  - algorithms,
  - memory requirements.

- **Complete picture** for this new framework.
## Results overview (2/2)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Reachability</td>
<td>$P$ [Put94]</td>
<td>$P(M) \cdot E(Q)$ [EKVY08], PSPACE-h</td>
<td>—</td>
</tr>
<tr>
<td>$f \in \mathcal{F}$</td>
<td>$P$ [CH09]</td>
<td>$P$</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h.</td>
</tr>
<tr>
<td>MP</td>
<td>$P$ [Put94]</td>
<td>$P$</td>
<td>$P$</td>
</tr>
<tr>
<td>MP</td>
<td>$P$ [Put94]</td>
<td>$P(M) \cdot E(Q)$</td>
<td>$P(M) \cdot E(Q)$</td>
</tr>
<tr>
<td>SP</td>
<td>$P(M) \cdot P_{ps}(Q)$ [HK14] PSPACE-h. [HK14]</td>
<td>$P(M) \cdot P_{ps}(Q)$ (one target) PSPACE-h. [HK14]</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h. [HK14]</td>
</tr>
<tr>
<td>$\varepsilon$-gap DS</td>
<td>$P_{ps}(M, Q, \varepsilon)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ PSPACE-h.</td>
</tr>
</tbody>
</table>

- $\mathcal{F} = \{\text{inf, sup, lim inf, lim sup}\}$
- $M = \text{model size, } Q = \text{query size}$
- $P(x), E(x)$ and $P_{ps}(x)$ resp. denote polynomial, exponential and pseudo-polynomial time in parameter $x$.

**All results without reference are new.**
Results overview (2/2)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Reachability</td>
<td>P ([\text{Put94}])</td>
<td>(P(M) \cdot E(Q)) ([\text{EKVY08}]), PSPACE-h</td>
<td>—</td>
</tr>
<tr>
<td>(f \in \mathcal{F})</td>
<td>P ([\text{CH09}])</td>
<td>P</td>
<td>(P(M) \cdot E(Q)) PSPACE-h</td>
</tr>
<tr>
<td>(\text{MP})</td>
<td>P ([\text{Put94}])</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>(\text{MP})</td>
<td>P ([\text{Put94}])</td>
<td>(P(M) \cdot E(Q))</td>
<td>(P(M) \cdot E(Q))</td>
</tr>
<tr>
<td>(\text{SP})</td>
<td>(P(M) \cdot P_{\text{ps}}(Q)) ([\text{HK14}]) PSPACE-h. ([\text{HK14}])</td>
<td>(P(M) \cdot P_{\text{ps}}(Q)) (one target) PSPACE-h. ([\text{HK14}])</td>
<td>(P(M) \cdot E(Q)) PSPACE-h. ([\text{HK14}])</td>
</tr>
<tr>
<td>(\varepsilon)-gap DS</td>
<td>(P_{\text{ps}}(M, Q, \varepsilon)) NP-h.</td>
<td>(P_{\text{ps}}(M, \varepsilon) \cdot E(Q)) NP-h.</td>
<td>(P_{\text{ps}}(M, \varepsilon) \cdot E(Q)) PSPACE-h.</td>
</tr>
</tbody>
</table>

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.
### Results overview (2/2)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reachability</strong></td>
<td>$P$ [Put94]</td>
<td>$P(M) \cdot E(Q)$ [EKVY08], PSPACE-h</td>
<td>—</td>
</tr>
<tr>
<td>$f \in F$</td>
<td>$P$ [CH09]</td>
<td>$P$</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h.</td>
</tr>
<tr>
<td>MP</td>
<td>$P$ [Put94]</td>
<td>$P$</td>
<td>$P$</td>
</tr>
<tr>
<td>MP</td>
<td>$P$ [Put94]</td>
<td>$P(M) \cdot E(Q)$</td>
<td>$P(M) \cdot E(Q)$</td>
</tr>
<tr>
<td>SP</td>
<td>$P(M) \cdot P_{ps}(Q)$ [HK14] PSPACE-h. [HK14]</td>
<td>$P(M) \cdot P_{ps}(Q)$ (one target) PSPACE-h. [HK14]</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h. [HK14]</td>
</tr>
<tr>
<td>$\varepsilon$-gap DS</td>
<td>$P_{ps}(M, Q, \varepsilon)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ PSPACE-h.</td>
</tr>
</tbody>
</table>

**No time to discuss every result!**
## Results overview (2/2)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Reachability</td>
<td>$P$ [Put94]</td>
<td>$P(M) \cdot E(Q)$ [EKVY08], PSPACE-h</td>
<td>—</td>
</tr>
<tr>
<td>$f \in \mathcal{F}$</td>
<td>$P$ [CH09]</td>
<td>$P$</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h.</td>
</tr>
<tr>
<td>MP</td>
<td>$P$ [Put94]</td>
<td>$P$</td>
<td>$P$</td>
</tr>
<tr>
<td>MP</td>
<td>$P$ [Put94]</td>
<td>$P(M) \cdot E(Q)$</td>
<td>$P(M) \cdot E(Q)$</td>
</tr>
<tr>
<td>SP</td>
<td>$P(M) \cdot P_{ps}(Q)$ [HK14] PSPACE-h. [HK14]</td>
<td>$P(M) \cdot P_{ps}(Q)$ (one target) PSPACE-h. [HK14]</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h. [HK14]</td>
</tr>
<tr>
<td>$\varepsilon$-gap DS</td>
<td>$P_{ps}(M, Q, \varepsilon)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ PSPACE-h.</td>
</tr>
</tbody>
</table>

### 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!
## Results overview (2/2)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reachability</strong></td>
<td>P [Put94]</td>
<td>P(M)·E(Q) [EKVY08], PSPACE-h</td>
<td>—</td>
</tr>
<tr>
<td><strong>f ∈ F</strong></td>
<td>P [CH09]</td>
<td>P</td>
<td>P(M)·E(Q) PSPACE-h.</td>
</tr>
<tr>
<td><strong>MP</strong></td>
<td>P [Put94]</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td><strong>MP</strong></td>
<td>P [Put94]</td>
<td>P(M)·E(Q)</td>
<td>P(M)·E(Q)</td>
</tr>
<tr>
<td><strong>SP</strong></td>
<td>P(M)·P_{ps}(Q) [HK14] PSPACE-h. [HK14]</td>
<td>P(M)·P_{ps}(Q) (one target) PSPACE-h. [HK14]</td>
<td>P(M)·E(Q) PSPACE-h. [HK14]</td>
</tr>
<tr>
<td><strong>ε-gap DS</strong></td>
<td>P_{ps}(M, Q, ε) NP-h.</td>
<td>P_{ps}(M, ε)·E(Q) NP-h.</td>
<td>P_{ps}(M, ε)·E(Q) PSPACE-h.</td>
</tr>
</tbody>
</table>

### Four groups of results

2. **F and MP.** Easiest cases.
   - inf and sup: reduction to *multiple reachability*.
   - lim inf, lim sup and MP: *maximal end-component* (MEC) decomposition + reduction to multiple reachability.
### Results overview (2/2)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reachability</strong></td>
<td>$P([\text{Put}94])$</td>
<td>$P(M) \cdot E(Q) [\text{EKVY}08]$, PSPACE-h</td>
<td>$-$</td>
</tr>
<tr>
<td>$f \in F$</td>
<td>$P([\text{CH}09])$</td>
<td>$P$</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h.</td>
</tr>
<tr>
<td>MP</td>
<td>$P([\text{Put}94])$</td>
<td>$P$</td>
<td>$P$</td>
</tr>
<tr>
<td>MP</td>
<td>$P([\text{Put}94])$</td>
<td>$P(M) \cdot E(Q)$</td>
<td>$P(M) \cdot E(Q)$</td>
</tr>
<tr>
<td><strong>SP</strong></td>
<td>$P(M) \cdot P_{ps}(Q) [\text{HK}14]$ PSPACE-h. [\text{HK}14]</td>
<td>$P(M) \cdot P_{ps}(Q)$ (one target) PSPACE-h. [\text{HK}14]</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h. [\text{HK}14]</td>
</tr>
<tr>
<td>$\varepsilon$-gap DS</td>
<td>$P_{ps}(M, Q, \varepsilon)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ PSPACE-h.</td>
</tr>
</tbody>
</table>

### Four groups of results

#### 3. 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.
### Results overview (2/2)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reachability</strong></td>
<td>P [Put94]</td>
<td>P(M)·E(Q) [EKVY08], PSPACE-h</td>
</tr>
<tr>
<td>$f \in \mathcal{F}$</td>
<td>P [CH09]</td>
<td>P</td>
</tr>
<tr>
<td>MP</td>
<td>P [Put94]</td>
<td>P</td>
</tr>
<tr>
<td>MP</td>
<td>P [Put94]</td>
<td>P(M)·E(Q)</td>
</tr>
<tr>
<td><strong>SP</strong></td>
<td>P(M)·P_{ps}(Q) [HK14]</td>
<td>P(M)·P_{ps}(Q) (one target)</td>
</tr>
<tr>
<td><strong>E-gap DS</strong></td>
<td>P_{ps}(M, Q, \varepsilon)</td>
<td>P_{ps}(M, \varepsilon)·E(Q)</td>
</tr>
</tbody>
</table>

#### 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*.
**Results overview (2/2)**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reachability</strong></td>
<td>$P$ [Put94]</td>
<td>$P(M) \cdot E(Q)$ [EKVY08], PSPACE-h</td>
<td>—</td>
</tr>
<tr>
<td>$f \in F$</td>
<td>$P$ [CH09]</td>
<td>$P$</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h.</td>
</tr>
<tr>
<td><strong>MP</strong></td>
<td>$P$ [Put94]</td>
<td>$P$</td>
<td>$P$</td>
</tr>
<tr>
<td><strong>SP</strong></td>
<td>$P(M) \cdot P_{ps}(Q)$ [HK14] PSPACE-h. [HK14]</td>
<td>$P(M) \cdot P_{ps}(Q)$ (one target) PSPACE-h. [HK14]</td>
<td>$P(M) \cdot E(Q)$ PSPACE-h. [HK14]</td>
</tr>
<tr>
<td><strong>$\varepsilon$-gap DS</strong></td>
<td>$P_{ps}(M, Q, \varepsilon)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ NP-h.</td>
<td>$P_{ps}(M, \varepsilon) \cdot E(Q)$ PSPACE-h.</td>
</tr>
</tbody>
</table>

**Four groups of results**

1. **SP and DS.**
   - **Technical focus of this talk.**
     - Intuitive unfoldings, interesting tricks for DS.
     - Start simple and iteratively extend the solution.
Some related work

- **Same philosophy** (i.e., beyond uni-dimensional $E$ or $P$ maximization), $\neq$ approaches.
  - Beyond worst-case synthesis: $E +$ worst-case [BFRR14b].
  - Survey of recent extensions in VMCAI’15 [RRS15].
Some related work

- **Same philosophy** (i.e., beyond uni-dimensional $\mathbb{E}$ or $\mathbb{P}$ maximization), $\neq$ approaches.
  - Beyond worst-case synthesis: $\mathbb{E} +$ worst-case [BFRR14b].
  - Survey of recent extensions in VMCAI’15 [RRS15].

- Multi-dim. MDPs: DS [CMH06], MP [BBC+14, FKR95].
Some related work

- **Same philosophy** (i.e., beyond uni-dimensional $E$ or $P$ maximization), $\neq$ approaches.
  - Beyond worst-case synthesis: $E +$ worst-case [BFRR14b].
  - Survey of recent extensions in VMCAI’15 [RRS15].

- Multi-dim. MDPs: DS [CMH06], MP [BBC+14, FKR95].

- Many related works for each particular payoff: MP [Put94], SP [UB13, HK14], DS [Whi93, WL99, BCF+13], etc.
  - All with a *single* constraint.
Some related work

- **Same philosophy** (i.e., beyond uni-dimensional $E$ or $P$ maximization), $\neq$ approaches.


  - All with a *single* constraint.

- Multi-constraint percentile queries for LTL $[EKVY08]$.
  - Closest to our work.
  - We use *multiple reachability*. 
1. Context, MDPs, Strategies

2. Percentile Queries

3. Shortest Path

4. Discounted Sum

5. Conclusion
Single-constraint queries

Single-constraint percentile problem for SP

Given MDP $M = (S, A, \delta, w)$, initial state $s_{init}$, target set $T$, threshold $v \in \mathbb{N}$, and probability threshold $\alpha \in [0, 1] \cap \mathbb{Q}$, decide if there exists a strategy $\sigma$ such that $\mathbb{P}_{M, s_{init}}^{\sigma}[TS^T \leq v] \geq \alpha$.

▷ Hypothesis: all weights are non-negative.

Theorem

The above problem can be decided in pseudo-polynomial time and is PSPACE-hard. Optimal pure strategies with pseudo-polynomial memory exist and can be constructed in pseudo-polynomial time.

▷ Polynomial in the size of the MDP, but also in the threshold $v$.
▷ See [HK14] for hardness.
Pseudo-PTIME algorithm (1/2)

Key idea: pseudo-PTIME reduction to the stochastic reachability problem (SR - single target).
Pseudo-PTIME algorithm (1/2)

Key idea: pseudo-PTIME reduction to the \textit{stochastic reachability problem} (SR - single target).

\textbf{SR problem}

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

\textbf{Theorem}

The SR problem can be decided in \textit{polynomial time}. Optimal \textit{pure memoryless strategies} exist and can be constructed in polynomial time.

▷ Linear programming.
Sketch of the reduction

1. Start from $M$, $T = \{s_2\}$, and $v = 7$. 
Pseudo-PTIME algorithm (2/2)

Sketch of the reduction

1. Start from $M$, $T = \{s_2\}$, and $v = 7$.

2. Build $M_v$ by unfolding $M$, tracking the current sum up to the threshold $v$, and integrating it in the states of the expanded MDP.
Pseudo-PTIME algorithm (2/2)
Pseudo-PTIME algorithm (2/2)
Pseudo-PTIME algorithm (2/2)
Pseudo-PTIME algorithm (2/2)
Pseudo-PTIME algorithm (2/2)
Pseudo-PTIME algorithm (2/2)
Pseudo-PTIME algorithm (2/2)
Pseudo-PTIME algorithm (2/2)

3. Bijection between runs of $M$ and $M_v$

$$TS^T(\rho) \leq v \iff \rho' \models \Diamond T', \ T' = T \times \{0, 1, \ldots, v\}$$
Pseudo-PTIME algorithm (2/2)

3. **Bijection between runs of** $M$ **and** $M_v$

$$TS^T(\rho) \leq v \iff \rho' \models \Diamond T', T' = T \times \{0, 1, \ldots, v\}$$

4. **Solve the SR problem on** $M_v$

   - Memoryless strategy in $M_v \leadsto$ pseudo-polynomial memory in $M$ in general
Pseudo-PTIME algorithm (2/2)

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

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

For the single-constraint problem, both strategies are equivalent

we can discriminate them with richer queries
Multi-constraint queries (1/2)

Multi-constraint percentile problem for SP

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

$$Q := \bigwedge_{i=1}^{q} \mathbb{P}_{M, s_{\text{init}}}^{\sigma}[\text{TS}_{l_i}^{T_i} \leq v_i] \geq \alpha_i,$$

where $\text{TS}_{l_i}^{T_i}$ denotes the truncated sum on dimension $l_i$ and w.r.t. target set $T_i$. 
Multi-constraint queries (2/2)

Theorem

This problem can be decided in
- **exponential time** in general,
- **pseudo-polynomial time** for single-dimension single-target multi-constraint queries.

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

- Polynomial in the size of the MDP, blowup due to the query.
- Hardness already true for single-constraint [HK14].
- Wide extension for basically no price in complexity.

⚠️ Undecidable for arbitrary weights (2CM reduction)!
EXPTIME / pseudo-PTIME algorithm

1. Build an unfolded MDP $M_v$ similar to single-constraint case:
   - stop unfolding when all dimensions reach sum $v = \max_i v_i$. 
EXPTIME / pseudo-PTIME algorithm

1. Build an unfolded MDP $M_v$ similar to single-constraint case:
   - stop unfolding when all dimensions reach sum $v = \max_i v_i$.

2. Maintain single-exponential size by defining an equivalence relation between states of $M_v$:
   - $S_v \subseteq S \times (\{0, \ldots, v\} \cup \{\bot\})^d$,
   - pseudo-poly. if $d = 1$. 
EXPTIME / pseudo-PTIME algorithm

1. Build an unfolded MDP $M_v$ similar to single-constraint case:
   - stop unfolding when all dimensions reach sum $v = \max_i v_i$.

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

3. For each constraint $i$, compute a target set $R_i$ in $M_v$:
   - $\rho \models \text{constraint } i \text{ in } M \iff \rho' \models \Diamond R_i \text{ in } M_v$. 
EXPTIME / pseudo-PTIME algorithm

1. Build an unfolded MDP $M_v$ similar to single-constraint case:
   - stop unfolding when all dimensions reach sum $v = \max_i v_i$.

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

3. For each constraint $i$, compute a target set $R_i$ in $M_v$:
   - $\rho \models \text{constraint } i \text{ in } M \iff \rho' \models \Diamond R_i \text{ in } M_v$.

4. Solve a multiple reachability problem on $M_v$.
   - Generalizes the SR problem [EKVY08, RRS14].
   - Time polynomial in $M_v$ but exponential in $q$.
   - Single-dim. single target queries $\Rightarrow$ absorbing targets $\Rightarrow$ polynomial-time algorithm for multiple reachability.
1. Context, MDPs, Strategies

2. Percentile Queries

3. Shortest Path

4. Discounted Sum

5. Conclusion
Multi-constraint queries

Multi-constraint percentile problem for DS

Given \(d\)-dimensional MDP \(M = (S, A, \delta, w)\), initial state \(s_{\text{init}}\) 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 \(v_i \in \mathbb{N}\) and probability thresholds \(\alpha_i \in [0, 1] \cap \mathbb{Q}\), where \(i \in \{1, \ldots, q\}\), decide if there exists a strategy \(\sigma\) such that query \(Q\) holds, with

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

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

We allow arbitrary weights.
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].
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.
\( \varepsilon \)-gap percentile problem (1/3)

- Classical decision problem.
  - Two types of inputs: yes-inputs and no-inputs.
  - Correct answers required for both types.
\( \varepsilon \)-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 ε > 0.
ε-gap percentile problem (2/3)

We build an **algorithm**.

- **Inputs**: query \( Q \) 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 \( \sim \varepsilon \).
    
    \[ \Rightarrow \text{Incremental approximation scheme.} \]
ε-gap percentile problem (3/3)

Theorem

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

- Yes if there is a strategy satisfying query \( Q_{2 \cdot \varepsilon} \);
- No if there is no strategy satisfying query \( Q_{-2 \cdot \varepsilon} \);
- and arbitrarily otherwise.

▷ **Shifted query**: \( Q_x \equiv Q \) with value thresholds \( v_i + x \) (all other things being equal).
Theorem

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

- Yes if there is a strategy satisfying query \( Q_{2 \cdot \varepsilon} \);
- No if there is no strategy satisfying query \( Q_{-2 \cdot \varepsilon} \);
- and arbitrarily otherwise.

**Shifted query**: \( Q_x \equiv Q \) 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 [GJ79, BFRR14b]), 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 (≠ SP).
     - 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).
     - Not easy to know when to stop.
   - Use the **discount factor**.
     - Weights contribute less and less to the sum along a run.
     - The range of possible futures narrows the deeper we go.
     - 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, ≠ SP), too many possible label values.
   ▶ Introduce a rounding scheme of the numbers involved (inspired by [BCF+13]).
     ⇒ We bound the error due to cumulated roundings by $\varepsilon/2$.
     ⇒ Single-exponential width.
Algorithm: key ideas

1. Goal: multiple reachability over appropriate unfolding.

2. Pseudo-polynomial depth.


4. Leaf labels are off by at most $\varepsilon$. Classify each leaf w.r.t. each constraint.
   - Same idea as for SP.
     - Defining target sets for multiple reachability.
   - 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.


4. **Leaf labels are off by at most** $\varepsilon$. 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.
4. **Leaf labels are off by at most* $\varepsilon$.** 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 $\varepsilon$-gap problem.
1 Context, MDPs, Strategies

2 Percentile Queries

3 Shortest Path

4 Discounted Sum

5 Conclusion
Summary

- **Multi-constraint percentile queries.**
  - Generalizes the classical threshold probability problem.

- Wide range of payoffs: reachability, inf, sup, lim inf, lim sup, mean-payoff, shortest path, discounted sum.
  - Various techniques are needed.

- **Complexity usually acceptable.**
  - Often only polynomial in the model size, while exponential in the query size for the most general variants.
## Results overview

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reachability</strong></td>
<td>P [Put94]</td>
<td>P(M)·E(Q) [EKVV08], PSPACE-h</td>
<td>—</td>
</tr>
<tr>
<td>$f \in \mathcal{F}$</td>
<td>P [CH09]</td>
<td>P</td>
<td>P(M)·E(Q) PSPACE-h.</td>
</tr>
<tr>
<td>MP</td>
<td>P [Put94]</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>MP</td>
<td>P [Put94]</td>
<td>P(M)·E(Q)</td>
<td>P(M)·E(Q)</td>
</tr>
<tr>
<td>SP</td>
<td>P(M)·P_{ps}(Q) [HK14], PSPACE-h. [HK14]</td>
<td>P(M)·P_{ps}(Q) (one target) PSPACE-h. [HK14]</td>
<td>P(M)·E(Q) PSPACE-h. [HK14]</td>
</tr>
<tr>
<td>$\varepsilon$-gap DS</td>
<td>P_{ps}(M, Q, \varepsilon) NP-h.</td>
<td>P_{ps}(M, \varepsilon)·E(Q) NP-h.</td>
<td>P_{ps}(M, \varepsilon)·E(Q) PSPACE-h.</td>
</tr>
</tbody>
</table>

\[ \mathcal{F} = \{ \inf, \sup, \lim \inf, \lim \sup \} \]

\[ M = \text{model size}, \ Q = \text{query size} \]

\[ P(x), E(x) \text{ and } P_{ps}(x) \text{ resp. denote polynomial, exponential and pseudo-polynomial time in parameter } x. \]

Thank you! Any question?
References


Udi Boker and Thomas A. Henzinger. Exact and approximate determinization of discounted-sum automata. LMCS, 10(1), 2014.


References II

Krishnendu Chatterjee, Vojtech Forejt, and Dominik Wojtczak.
Multi-objective discounted reward verification in graphs and MDPs.

Boris V. Cherkassky, Andrew V. Goldberg, and Tomasz Radzik.
Shortest paths algorithms: Theory and experimental evaluation.

Krishnendu Chatterjee and Thomas A. Henzinger.
Probabilistic systems with limsup and liminf objectives.

Krishnendu Chatterjee, Rupak Majumdar, and Thomas A. Henzinger.
Markov decision processes with multiple objectives.

Krishnendu Chatterjee, Mickael Randour, and Jean-François Raskin.
Strategy synthesis for multi-dimensional quantitative objectives.

Kousha Etessami, Marta Z. Kwiatkowska, Moshe Y. Vardi, and Mihalis Yannakakis.
Multi-objective model checking of Markov decision processes.
Percentile performance criteria for limiting average markov decision processes.  

Michael R. Garey and David S. Johnson.  

Oded Goldreich.  
On promise problems: A survey.  

Christoph Haase and Stefan Kiefer.  
The odds of staying on budget.  

Martin L. Puterman.  

Mickael Randour, Jean-François Raskin, and Ocan Sankur.  
Percentile queries in multi-dimensional Markov decision processes.  

Mickael Randour, Jean-François Raskin, and Ocan Sankur.  
Variations on the stochastic shortest path problem.  
References IV

Stephen D. Travers.
The complexity of membership problems for circuits over sets of integers.

Michael Ummels and Christel Baier.
Computing quantiles in Markov reward models.

Douglas J. White.
Minimizing a threshold probability in discounted Markov decision processes.

Congbin Wu and Yuanlie Lin.
Minimizing risk models in Markov decision processes with policies depending on target values.