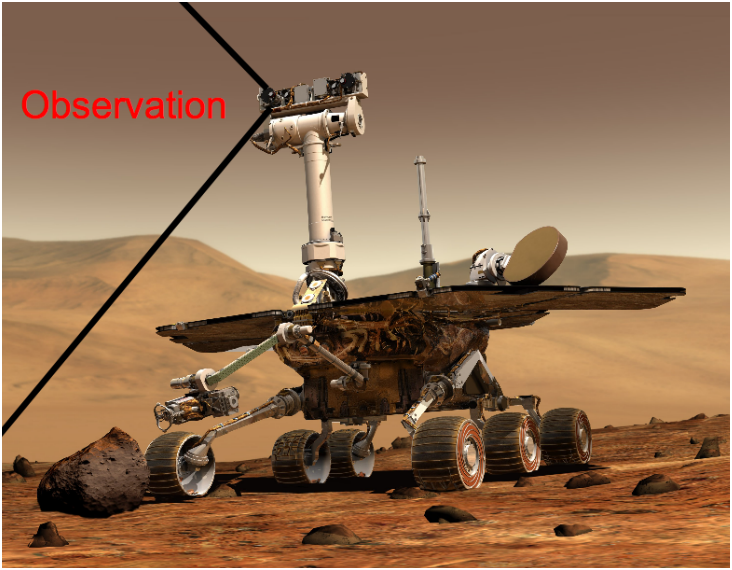
Optimistic Linear Support and Multi-objective POMDPs

Diederik M. Roijers
Informatics Institute
University of Amsterdam

in collaboration with:
Shimon Whiteson and Frans Oliehoek

October 20, 2015

Multiple objectives



Maximize coverage while minimizing damage

Outline

- Multi-objective decision problems
- Convex coverage sets

- Optimistic Linear Support
- Approximate single-objective solvers

- Multi-objective POMDPs
- OLS for MOPOMDPs
- Scalarized Perseus
- α-matrix reuse
- Experimental results

Do we need multi-objective models?

Sutton's Reward Hypothesis: "All of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received *scalar* signal (reward)."

Source: `http://rlai.cs.ualberta.ca/RLAI/rewardhypothesis.html`

- $V : \Pi \rightarrow \mathbb{R}$
- $V^\pi = E_\pi[\sum_t r_t]$
- $\pi^* = \max_\pi V^\pi$

Do we need multi-objective models?

Sutton's Reward Hypothesis: "All of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received *scalar* signal (reward)."

Source: `http://rlai.cs.ualberta.ca/RLAI/rewardhypothesis.html`

- $V : \Pi \to \mathbb{R}$
- $V^\pi = E_\pi[\sum_t r_t]$
- $\pi^* = \max_\pi V^\pi$

Do we need multi-objective models?

Sutton's Reward Hypothesis: "All of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received *scalar* signal (reward)."

Source: `http://rlai.cs.ualberta.ca/RLAI/rewardhypothesis.html`

- $V : \Pi \to \mathbb{R}$
- $V^\pi = E_\pi[\sum_t r_t]$
- $\pi^* = \max_\pi V^\pi$

Why Multi-Objective Decision Making?

- *The weak argument*: real-world problems are multi-objective!

$$\mathbf{V} : \Pi \to \mathbb{R}^n$$

- Objection: why not just *scalarize*?
- Scalarization function projects multi-objective value to a scalar:

$$V^\pi_{\mathbf{w}} = f(\mathbf{V}^\pi, \mathbf{w})$$

- Linear case:

$$V^\pi_{\mathbf{w}} = \sum_{i=1}^{n} w_i V^\pi_i = \mathbf{w} \cdot \mathbf{V}^\pi$$

- A priori prioritization of the objectives
- *The weak argument is necessary but not sufficient*

Why Multi-Objective Decision Making?

- *The weak argument*: real-world problems are multi-objective!

$$\mathbf{V} : \Pi \rightarrow \mathbb{R}^n$$

- Objection: why not just *scalarize*?
- Scalarization function projects multi-objective value to a scalar:

$$V_{\mathbf{w}}^{\pi} = f(\mathbf{V}^{\pi}, \mathbf{w})$$

- Linear case:

$$V_{\mathbf{w}}^{\pi} = \sum_{i=1}^{n} w_i V_i^{\pi} = \mathbf{w} \cdot \mathbf{V}^{\pi}$$

- A priori prioritization of the objectives
- *The weak argument is necessary but not sufficient*

Why Multi-Objective Decision Making?

- *The weak argument*: real-world problems are multi-objective!

$$\mathbf{V} : \Pi \to \mathbb{R}^n$$

- Objection: why not just *scalarize*?
- Scalarization function projects multi-objective value to a scalar:

$$V_{\mathbf{w}}^{\pi} = f(\mathbf{V}^{\pi}, \mathbf{w})$$

- Linear case:

$$V_{\mathbf{w}}^{\pi} = \sum_{i=1}^{n} w_i V_i^{\pi} = \mathbf{w} \cdot \mathbf{V}^{\pi}$$

- A priori prioritization of the objectives
- *The weak argument is necessary but not sufficient*

Why Multi-Objective Decision Making?

- *The weak argument*: real-world problems are multi-objective!

$$\mathbf{V} : \Pi \to \mathbb{R}^n$$

- Objection: why not just *scalarize*?
- Scalarization function projects multi-objective value to a scalar:

$$V_{\mathbf{w}}^{\pi} = f(\mathbf{V}^{\pi}, \mathbf{w})$$

- Linear case:

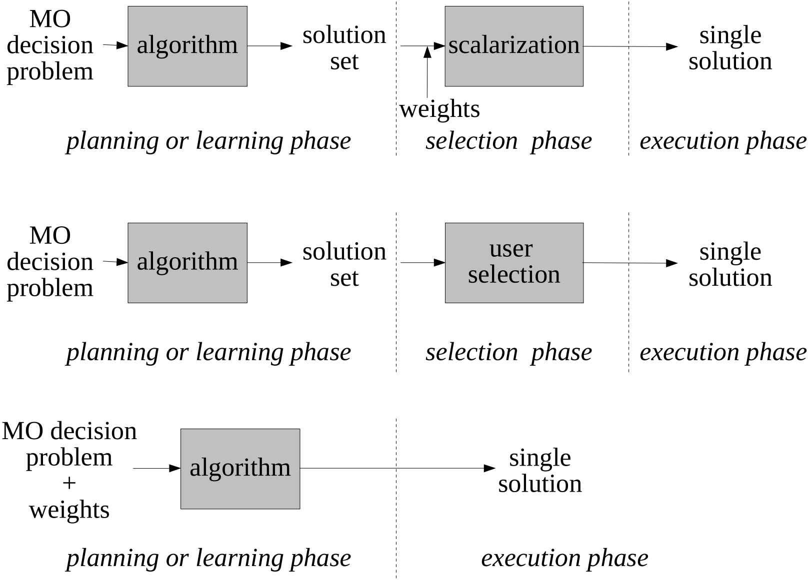
$$V_{\mathbf{w}}^{\pi} = \sum_{i=1}^{n} w_i V_i^{\pi} = \mathbf{w} \cdot \mathbf{V}^{\pi}$$

- A priori prioritization of the objectives
- *The weak argument is necessary but not sufficient*

Why Multi-Objective Decision Making?

- *The strong argument*: a priori scalarization is sometimes impossible, infeasible, or undesirable
- Instead produce the *coverage set* of undominated solutions
- Three scenario's

Motivating scenarios



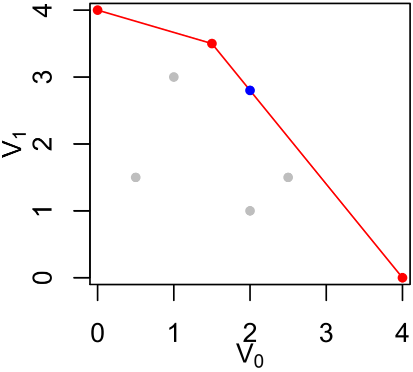
Utility-based approach

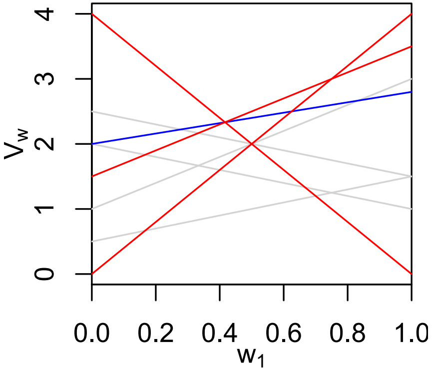
- Scalarization is *explicit* or *implicit*, but always happens
- Scalarization function: $V_{\mathbf{w}} = f(\mathbf{V}, \mathbf{w})$
- Choose the solution set by:
  - What do we know about f?
  - Stochastic policies allowed?
  - Non-stationary policies allowed?

Utility-based approach

- Scalarization is *explicit* or *implicit*, but always happens
- Scalarization function: $V_{\mathbf{w}} = f(\mathbf{V}, \mathbf{w})$
- Choose the solution set by:
  - What do we know about f?
  - Stochastic policies allowed?
  - Non-stationary policies allowed?

Convex coverage set (CCS)





- Scalarization function: $f(\mathbf{V}^\pi, \mathbf{w}) = \mathbf{w} \cdot \mathbf{V}^\pi$
- Scalarized value function: $V^*_{CCS}(\mathbf{w}) = \max_\pi \mathbf{w} \cdot \mathbf{V}^\pi$
- Piece-wise linear and convex (PWLC) function

Problem Taxonomy

	single policy (known weights)		*multiple policies (unknown weights or decision support)*	
	deterministic	stochastic	deterministic	stochastic
linear scalarization	one deterministic stationary policy		*convex coverage set of deterministic stationary policies*	
monotonically increasing scalarization	one deterministic non-stationary policy	one mixture policy of two or more deterministic stationary policies	Pareto coverage set of deterministic non-stationary policies	*convex coverage set of deterministic stationary policies*

Outline

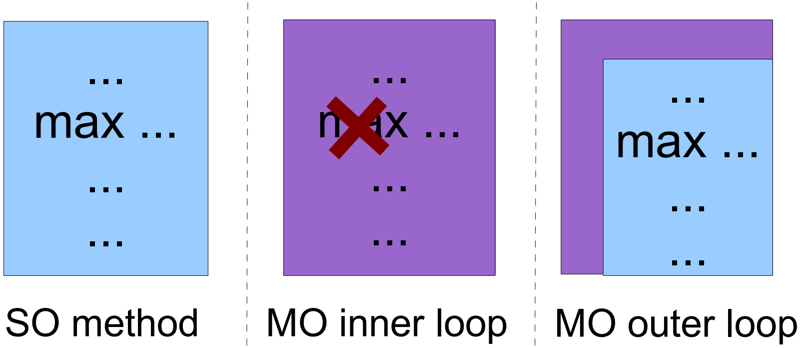
- Multi-objective decision problems
- Convex coverage sets

- Optimistic linear support
- Approximate single-objective solvers

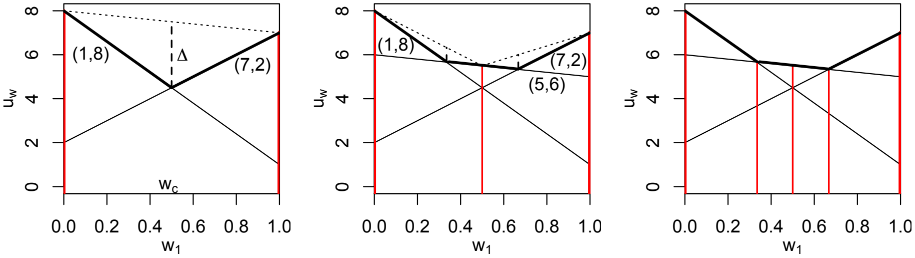
- Multi-objective POMDPs
- OLS for MOPOMDPs
- Scalarized Perseus
- α-matrix reuse
- Experimental results

Optimistic linear support

- Optimistic Linear Support (OLS)
- *Outer loop* approach: series *scalarized* instances with different **w**

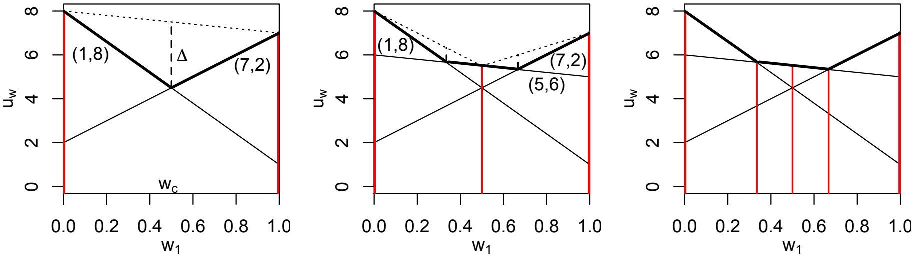


Optimistic linear support



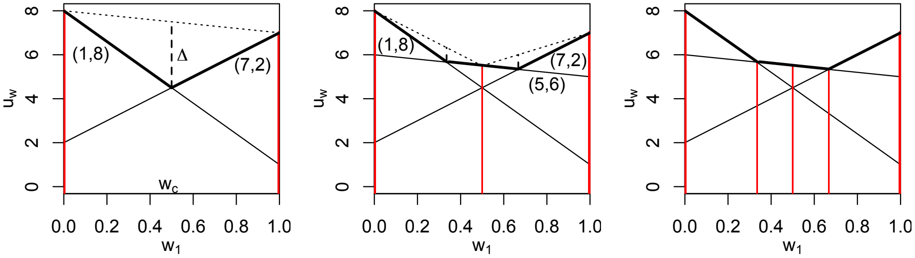
- Terminates after checking only a finite number of weights **w**
- Exact solutions if single-objective solver is exact
- Anytime

Optimistic linear support



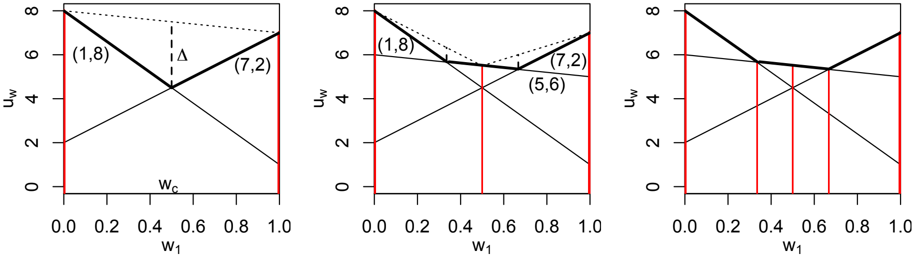
- Terminates after checking only a finite number of weights **w**
- Exact solutions if single-objective solver is exact
- Anytime

Optimistic linear support



- Terminates after checking only a finite number of weights **w**
- Exact solutions if single-objective solver is exact
- Anytime

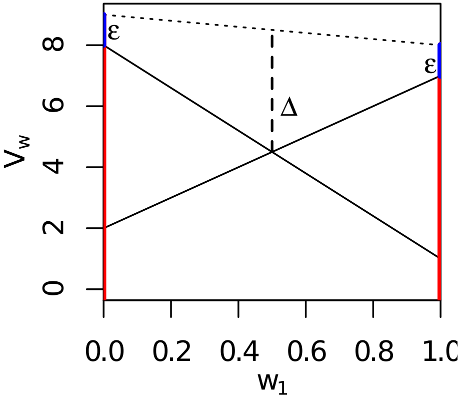
Optimistic linear support



- Terminates after checking only a finite number of weights **w**
- Exact solutions if single-objective solver is exact
- Anytime

Optimistic linear support

- ε-approximate single-objective solver
- OLS produces an *ε-CCS*



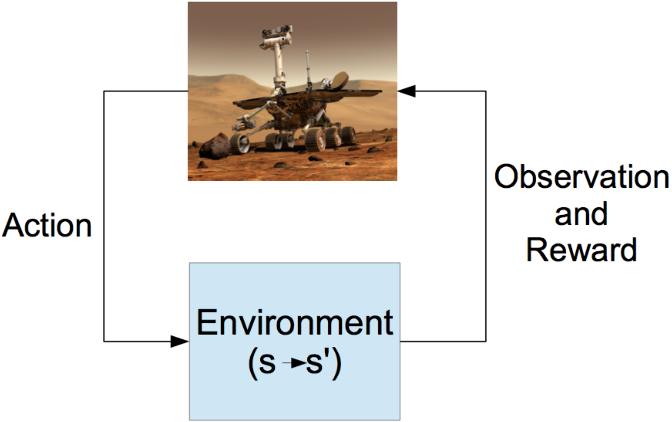
Outline

- Multi-objective decision problems
- Convex coverage sets

- Optimistic linear support
- Approximate single-objective solvers

- Multi-objective POMDPs
- OLS for MOPOMDPs
- Scalarized Perseus
- α-matrix reuse
- Experimental results

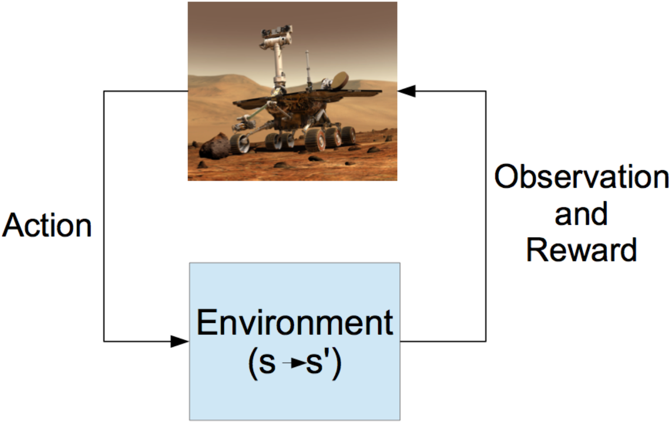
Multi-objective Partially Observable MDPs



- Multiple objectives
- Vector-valued policy values
- Set of all possibly optimal policies

$$V_{\mathbf{w}}^{\pi} = \mathbf{w} \cdot \mathbf{V}^{\pi} = w_1 V_{coverage}^{\pi} + w_2 V_{damage}^{\pi}$$

Multi-objective Partially Observable MDPs



- Multiple objectives
- Vector-valued policy values
- Set of all possibly optimal policies

$$V_{\mathbf{w}}^{\pi} = \mathbf{w} \cdot \mathbf{V}^{\pi} = w_1 V_{coverage}^{\pi} + w_2 V_{damage}^{\pi}$$

Approach

- Optimistic Linear Support
- Opens way to efficient MOPOMDP planning
- Solve as series of scalarized POMDPs
- Point-based POMDP planners
- Smart choices of scalarized instances

Optimistic linear support for POMDPs

- Point-based methods represent value by α-vectors

$$\alpha = \begin{pmatrix} V(s_1) \\ V(s_2) \\ V(s_3) \\ V(s_4) \end{pmatrix}$$

- $V^{\alpha}(b_0) = b_0 \cdot \alpha$

- Adapt point-based methods to return α-matrices

$$A = \begin{array}{c} \begin{array}{cc} obj\ 1: & obj\ 2: \end{array} \\ \begin{pmatrix} V_1(s_1) & V_2(s_1) \\ V_1(s_2) & V_2(s_2) \\ V_1(s_3) & V_2(s_3) \\ V_1(s_4) & V_2(s_4) \end{pmatrix} \end{array}$$

- $\mathbf{V}^{A}(b_0) = b_0 A$

- Adapted point-based backups

Optimistic linear support for POMDPs

- Point-based methods represent value by α-vectors

$$\alpha = \begin{pmatrix} V(s_1) \\ V(s_2) \\ V(s_3) \\ V(s_4) \end{pmatrix}$$

- $V^{\alpha}(b_0) = b_0 \cdot \alpha$

- Adapt point-based methods to return α-matrices

$$A = \begin{array}{c} \begin{array}{cc} obj\ 1: & obj\ 2: \end{array} \\ \begin{pmatrix} V_1(s_1) & V_2(s_1) \\ V_1(s_2) & V_2(s_2) \\ V_1(s_3) & V_2(s_3) \\ V_1(s_4) & V_2(s_4) \end{pmatrix} \end{array}$$

- $\mathbf{V}^{A}(b_0) = b_0 A$
- Adapted point-based backups

Multi-objective Point-based backups

Back-projection of α-vectors $\alpha_i \in \mathcal{A}_k$:

$$g_i^{a,o}(s) = \sum_{s' \in S} O(a, s', o) T(s, a, s') \alpha_i(s')$$

$$\alpha_{k+1}^{b,a} = r^a + \gamma \sum_{o \in \Omega} \arg\max_{g^{a,o}} b \cdot g^{a,o}$$

$$\texttt{backup}(\mathcal{A}_k, b) = \arg\max_{\alpha_{k+1}^{b,a}} b \cdot \alpha_{k+1}^{b,a}$$

Back-projection of α-matrices $\mathbf{A}_i \in \mathcal{A}_k$, for a *given* $\mathbf{w}$:

$$\mathbf{G}_i^{a,o}(s) = \sum_{s' \in S} O(a, s', o) T(s, a, s') \mathbf{A}_i(s')$$

$$\mathbf{A}_{k+1}^{b,a} = r^a + \gamma \sum_{o \in \Omega} \arg\max_{G^{a,o}} b\ \mathbf{G}^{a,o} \mathbf{w}$$

$$\texttt{backupMO}(\mathcal{A}_k, b, \mathbf{w}) = \arg\max_{\mathbf{A}_{k+1}^{b,a}} b \mathbf{A}_{k+1}^{a,b} \mathbf{w}$$

Multi-objective Point-based backups

Back-projection of α-vectors $\alpha_i \in \mathcal{A}_k$:

$$g_i^{a,o}(s) = \sum_{s' \in S} O(a, s', o) T(s, a, s') \alpha_i(s')$$

$$\alpha_{k+1}^{b,a} = r^a + \gamma \sum_{o \in \Omega} \arg\max_{g^{a,o}} b \cdot g^{a,o}$$

$$\texttt{backup}(\mathcal{A}_k, b) = \arg\max_{\alpha_{k+1}^{b,a}} b \cdot \alpha_{k+1}^{b,a}$$

Back-projection of α-matrices $\mathbf{A}_i \in \mathcal{A}_k$, for a *given* $\mathbf{w}$:

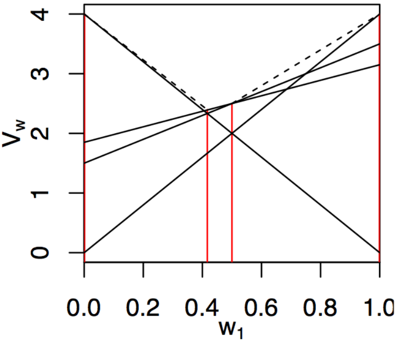
$$\mathbf{G}_i^{a,o}(s) = \sum_{s' \in S} O(a, s', o) T(s, a, s') \mathbf{A}_i(s')$$

$$\mathbf{A}_{k+1}^{b,a} = r^a + \gamma \sum_{o \in \Omega} \arg\max_{G^{a,o}} b \ \mathbf{G}^{a,o}\mathbf{w}$$

$$\texttt{backupMO}(\mathcal{A}_k, b, \mathbf{w}) = \arg\max_{\mathbf{A}_{k+1}^{b,a}} b\mathbf{A}_{k+1}^{a,b}\mathbf{w}$$

Optimistic linear support with alpha reuse

- Starting from scratch for each **w** is inefficient
- Intuition: when **w** and **w**$'$ are close, so are the optimal policies and values
- Hot start point-based planner using α-matrices
- More and more effective as **w**'s lie closer together



Theoretical results

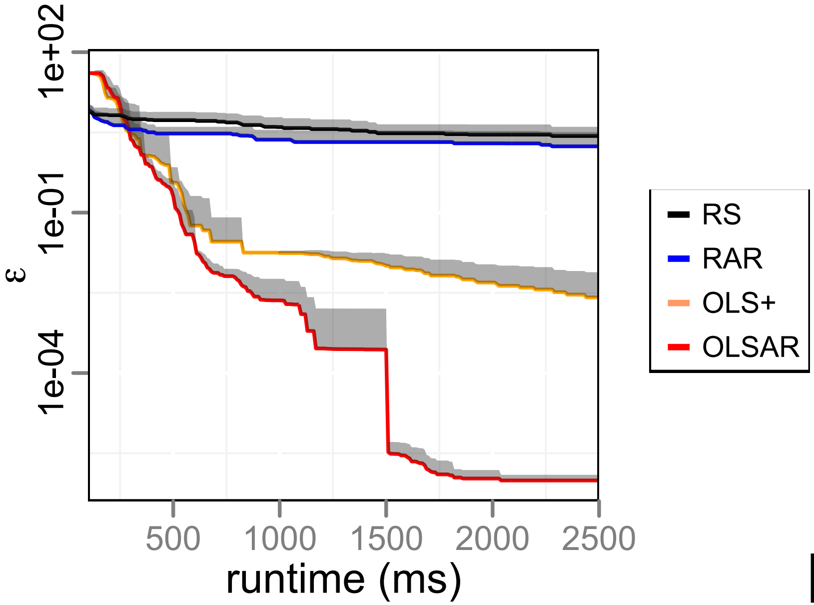
Theorem

OLSAR requires a finite number of calls to the point-based solver to converge.

Theorem

OLSAR produces an ε-approximate solution set.
ε is inherited from the single-objective method.

Sample of results: 3-objective tiger



Conclusions

- Use point-based methods for MOPOMDPs
- First method that reasonably scales
- Bounded approximation
- Alpha reuse is key to keeping MOPOMDPs tractable

$\texttt{OLS}(m, \texttt{SolveSingleObjective}, \epsilon)$ // Without reuse

$S \leftarrow \emptyset$ //partial CCS
$Q \leftarrow$ an empty priority queue
foreach *extremum of the weight simplex* $\mathbf{w}_e$ **do**
    Q.add($\mathbf{w}_e$, ∞) // add extrema with infinite priority
while $\neg$*Q.isEmpty()* $\wedge$ $\neg$*timeOut* **do**
    $\mathbf{w} \leftarrow$ Q.pop()
    $\mathbf{V} \leftarrow \texttt{SolveSingleObjective}(m, \mathbf{w})$
    if $\mathbf{V} \notin S$ **then**
        $S \leftarrow S \cup \{\mathbf{V}\}$
        delete obsolete corner weights from Q
        $W_{\mathbf{V}} \leftarrow$ the new corner weights that involve $\mathbf{V}$
        foreach $\mathbf{w} \in W_{\mathbf{V}}$ **do**
            $\Delta_r(\mathbf{w}) \leftarrow$ max. possible rel. improvement at $\mathbf{w}$
            if $\Delta_r(\mathbf{w}) > \epsilon$ **then**
                Q.add($\mathbf{w}$, $\Delta_r(\mathbf{w})$)
return *S and the highest* $\Delta_r(\mathbf{w})$ *left in* Q

$\mathcal{A}' \leftarrow \mathcal{A}$;
$\mathcal{A} \leftarrow \{-\vec{\infty}\}$; `// worst possible vector in a singleton set`
while $\max_b \max_{\mathbf{A}' \in \mathcal{A}'} b\mathbf{A}'\mathbf{w} - (\max_{\mathbf{A} \in \mathcal{A}} b\mathbf{A}\mathbf{w}) > \eta$ **do**
  $\mathcal{A} \leftarrow \mathcal{A}'$; $\mathcal{A}' \leftarrow \emptyset$; $B' \leftarrow B$;
  while $B' \neq \emptyset$ **do**
    Randomly select b from B';
    $\mathbf{A} \leftarrow \texttt{backupMO}(\mathcal{A}, b, \mathbf{w})$;
    $\mathcal{A}' \leftarrow \mathcal{A}' \cup \{ \underset{\mathbf{A}' \in (\mathcal{A} \cup \{\mathbf{A}\})}{\arg\max} \; b\mathbf{A}'\mathbf{w}\}$;
    $B' \leftarrow \{b \in B' : \max_{\mathbf{A}' \in \mathcal{A}'} b\mathbf{A}'\mathbf{w} < \max_{\mathbf{A} \in \mathcal{A}} b\mathbf{A}\mathbf{w}\}$;

return $\mathcal{A}'$;

$\texttt{OLSAR}(b_0, \eta)$ // With Reuse

$X \leftarrow \emptyset$; `// partial CCS of multi-objective value vectors` $\mathbf{V}_{b_0}$
$WV_{old} \leftarrow \emptyset$; `// searched weights and scalarized values`
$Q \leftarrow$ priority queue with weights to search;
Add extrema of the weight simplex to Q with infinite priority;
$\mathcal{A}_{all} \leftarrow$ a set of α-matrices forming a lower bound on the value;
$B \leftarrow$ set of sampled belief points (e.g., by random exploration);
while $\neg Q.\texttt{isEmpty}() \wedge \neg timeOut$ **do**
- $\mathbf{w} \leftarrow Q.\texttt{dequeue}()$; `// Retrieve a weight vector`
- $\mathcal{A}_r \leftarrow$ select the best $\mathbf{A}$ from $\mathcal{A}_{all}$ for each $b \in B$, given $\mathbf{w}$;
- $\mathcal{A}_{\mathbf{w}} \leftarrow \texttt{solveScalarizedPOMDP}(\mathcal{A}_r, B, \mathbf{w}, \eta)$;
- $\mathbf{V}_{b_0} \leftarrow \max_{\mathbf{A} \in \mathcal{A}_{\mathbf{w}}} b_0 \mathbf{A} \mathbf{w}$;
- $\mathcal{A}_{all} \leftarrow \mathcal{A}_{all} \cup \mathcal{A}_{\mathbf{w}}$;
- $WV_{old} = WV_{old} \cup \{(\mathbf{w},\ \mathbf{w} \cdot \mathbf{V}_{b_0})\}$;
- **if** $\mathbf{V}_{b_0} \notin X$ **then**
  - $X \leftarrow X \cup \{\mathbf{V}_{b_0}\}$;
  - $W \leftarrow$ compute new corner weights and maximum possible improvements $(\mathbf{w}, \Delta_{\mathbf{w}})$ using WV_{old} and X;
  - $Q.\texttt{addAll}(W)$;

return X;

Cheng's theorem

Theorem

(Cheng 1988) The maximum value of:

$$\max_{\mathbf{w},\mathbf{u} \in CCS} \min_{\mathbf{v} \in S} \mathbf{w} \cdot \mathbf{u} - \mathbf{w} \cdot \mathbf{v},$$

i.e., the maximal improvement to S by adding a vector to it, is at one of the corner weights.

Optimistic CCS

Definition

An *optimistic hypothetical CCS*, $\overline{CCS}$ is a set of payoff vectors that yields the highest possible scalarized value for all possible $\mathbf{w}$ consistent with finding the vectors S at the weights in $\mathcal{W}$.

For a given $\mathbf{w}$, the scalarized value of $u^*_{\overline{CCS}}(\mathbf{w})$ can be found by solving the following linear program:

$$\max \ \mathbf{w} \cdot \mathbf{v}$$
$$\text{subject to} \quad \mathcal{W}\mathbf{v} \leq \mathbf{u}^*_{S,\mathcal{W}},$$

where $\mathbf{u}^*_{S,\mathcal{W}}$ is a vector containing $u^*_S(\mathbf{w}')$ for all $\mathbf{w}' \in \mathcal{W}$.