Heuristic Search for Multi-Objective Probabilistic Planning

Dillon Chen,[1] Felipe Trevizan,[1] Sylvie Thiébaux[1,2]

[1]School of Computing, The Australian National University

[2]LAAS-CNRS, ANITI, Université de Toulouse

{Dillon.Chen, Felipe.Trevizan, Sylvie.Thiebaux}@anu.edu.au

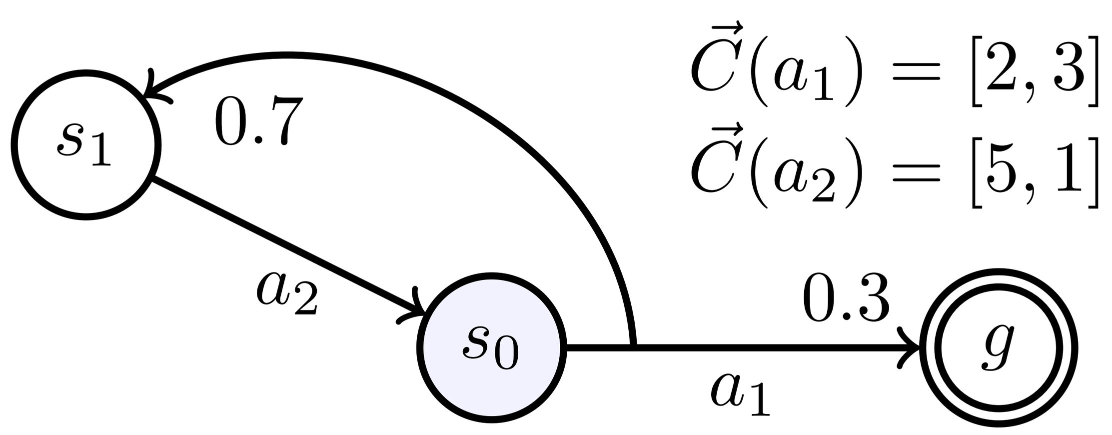
Contributions

- formalism for multi-objective probabilistic planning: **MOSSP**
- heuristic search algorithms for solving MOSSPs: **(i)MOLAO***, **MOLRTDP**
- new MOSSP heuristics
- assumptions for convergence of MOVI for MOSSPs

MOSSP

- A *multi-objective stochastic shortest path problem* (***MOSSP***) is a tuple $(S, s_0, G, A, P, \vec{C})$ where:
  - S is a finite set of states
  - s_0 is an initial state
  - $G \subseteq S$ is a set of goal states,
  - A is a finite set of actions,
  - $P(s' \mid s, a)$ is the probability of reaching s' after applying action a in s, and
  - $\vec{C}(a) \in \mathbb{R}^n_{\geq 0}$ is the n-dimensional vector representing the cost of action a.
- generalises planning problems to involve:

1. multiple objectives (MO)
2. stochastic actions (SSP)



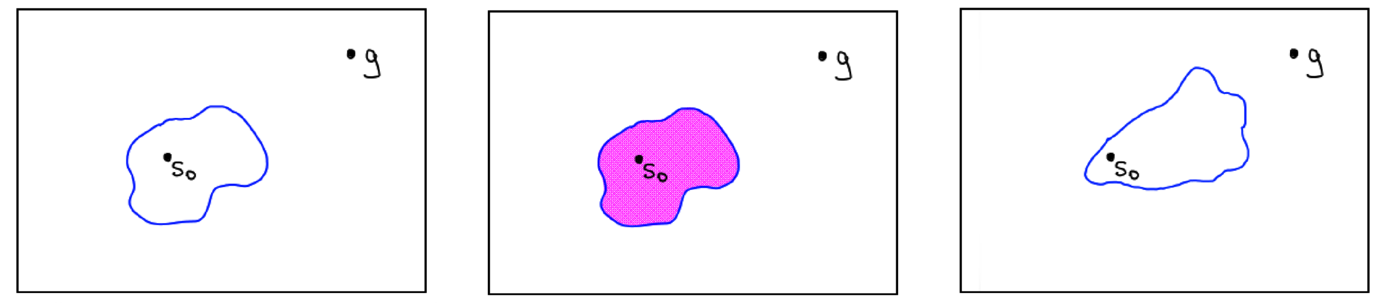
Heuristic Search

- heuristic search powerful for (optimal) planning
- heuristic search algorithms:

	SO	MO
determinstic	A*, BiA* etc.	NAMOA*
stochastic	(i)LAO*, LRTDP etc.	**(i)MOLAO***, **MOLRTDP**

iMOLAO*

```
Algorithm: iMOLAO*
Data: MOSSP problem P = (S, s0, G, A, P, C⃗), heuristic H, and
      consistency threshold ε
V ← H; Π ← ∅; F ← {s0} ; I ← ∅; N ← {s0}
while ((F ∩ N) \ G ≠ ∅) ∧ (max_{s∈N} res(s) < ε) do
    F = ∅
    // Step 1.
    N ← postorderTraversalDFS(s_I, Π)
    // Step 2.
    for s ∈ N in the computed order do
        V(s) ← BellmanBackup(s)
        // Step 3.
        Π(s) = getActions(s, V)
        if s ∉ I then F = F ∪ {s} ;
        I = I ∪ {s}
return V
```



1. collect states N of the current best partial solution Π in postorder traversal DFS order

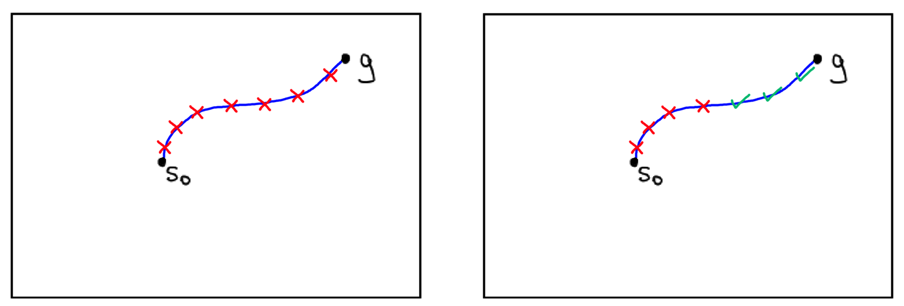
2. run MO Bellman Backups once on each state in the collected order

3. update new current partial solution Π; go to 1. and repeat until covergence criteria met

MOLRTDP

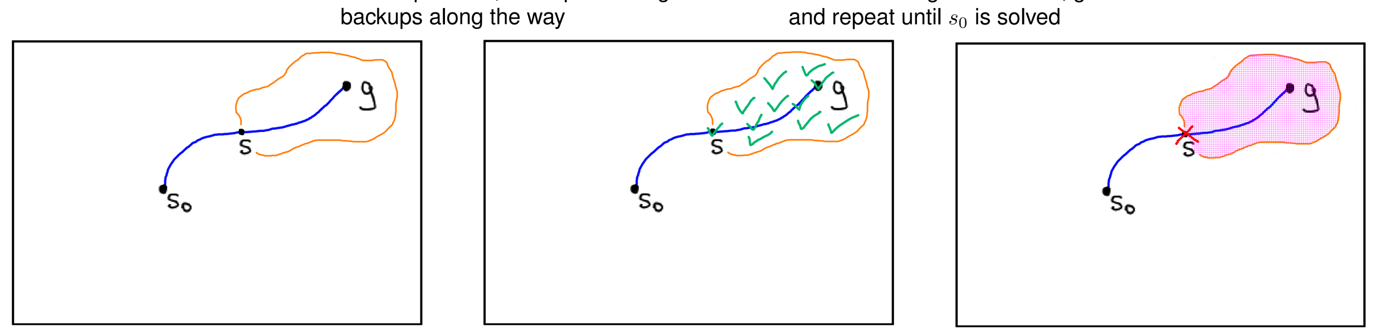
```
Algorithm: MOLRTDP
Data: MOSSP problem
      P = (S, s0, G, A, P, C⃗), heuristic H,
      and consistency threshold ε
procedure MOLRTDP(P, ε, H)
    V ← H
    while ¬s0.solved do
        visited ← ∅
        s ← s0
        // Step 1.
        while ¬s.solved do
            visited.push(s)
            if s ∈ G then break ;
            V(s) ← BellmanBackup(s)
            a ← sampleUnsolvedGreedyAction(s)
            s ← sampleUnsolvedNextState(s, a)
        // Step 2.
        while ¬visited.empty() do
            s ← visited.pop()
            if ¬checkSolved(s) then break ;
    return V
```

```
routine checkSolved(s)
    rv ← true; open ← ∅; closed ← ∅
    if ¬s.solved then open.push(s) ;
    // Step 2.1.
    while ¬open.empty() do
        s ← open.pop()
        if res(s) > ε then
            rv ← false
            continue
        for a ∈ getActions(s, V) do
            for s' ∈ successors(s, a) do
                if ¬s'.solved ∧ s' ∉ open ∪ closed
                  then open.push(s') ;
    // Step 2.2.a.
    if rv then for s ∈ closed do
      s.solved = true ;
    // Step 2.2.b.
    else
        while closed ≠ ∅ do
            s ← closed.pop()
            V(s) ← BellmanBackup(s)
    return rv
```



1. random greedy walk to the goal on unsolved states if possible, while performing backups along the way

2. update if states are solved in reverse order of the trialled walk using checkSolved; go to 1. and repeat until s_0 is solved



2.1. collect all greedy ancestor states of s under the current partial solution

2.2.a. if ancestor state values have converged, set state and ancestors as solved

2.2.b. else run backups on the state and all its ancestors

MOSSP heuristics

- MOSSP heuristic for a state is a finite set of vectors $\mathbf{H}(s) \subset \mathbb{R}^n_{\geq 0}$
- $\mathbf{H}$ is an *admissible heuristic* if $\forall s \in S \setminus G$, for all $\vec{v} \in \mathbf{V}^*(s)$ there exists $\vec{u} \in \mathbf{H}(s)$ such that $\vec{u} \preceq \vec{v}$ where $\mathbf{V}^*$ is the optimal value function, and $\forall g \in G, \mathbf{H}(g) = \{\vec{0}\}$.

Domain independent MOSSP heuristics

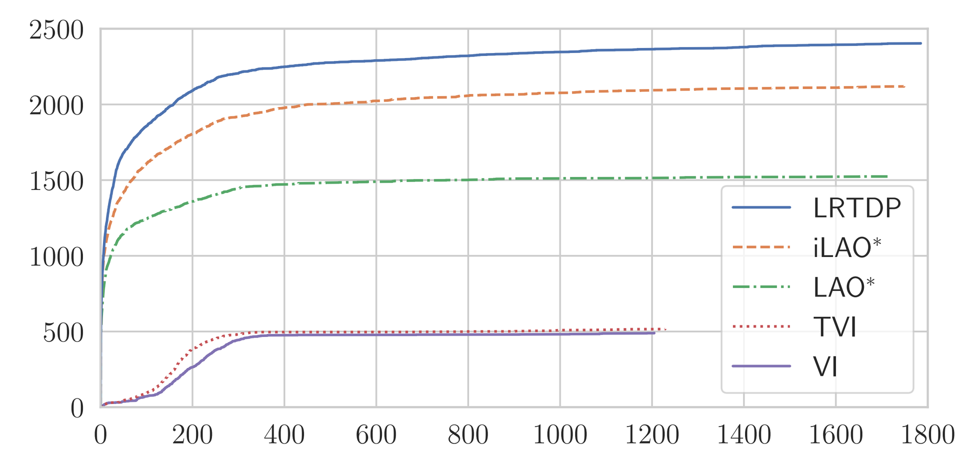
- no heuristic: the zero heuristic defined by $\mathbf{H}_{zero}(s) = \{\vec{0}\}$
- can leverage (stochastic) SO heuristics
  - apply an SO heuristic h_i to each objective in isolation, resulting in a single vector: $\mathbf{H}_{ideal}(s) = \{[h_1(s), \ldots h_n(s)]\}$
- can leverage (deterministic) MO heuristics
  - construct determinised problem by replacing each probabilistic effect with a deterministic action
- can use abstraction heuristics
  - combine values from solving smaller projections of the problem

	SO	MO
determinstic	$\mathbf{H}_{zero}$, $\mathbf{H}^{max}_{ideal}$	$\mathbf{H}^{comax}_{mo}$, $\mathbf{H}^{pdb2}_{mo}$, $\mathbf{H}^{pdb3}_{mo}$
stochastic	$\mathbf{H}^{pdb2}_{ideal}$, $\mathbf{H}^{pdb3}_{ideal}$	$\mathbf{H}^{pdb2}_{mossp}$, $\mathbf{H}^{pdb3}_{mossp}$

Experimental results

5 planners, 10 heuristics, 7 domains, 610 problems;

$5 \times 10 \times 610 = 30500$ possible experimental configurations



Cumulative coverage per planner, marginalised by heuristics. MO prefix omitted.

Heuristic	Coverage
$\mathbf{H}^{pdb3}_{mossp}$	893.5
$\mathbf{H}^{pdb2}_{mossp}$	893.3
$\mathbf{H}^{pdb3}_{mo}$	871.2
$\mathbf{H}^{pdb2}_{mo}$	851.8
$\mathbf{H}^{pdb3}_{ideal}$	768.7
$\mathbf{H}^{max}_{ideal}$	755.2
$\mathbf{H}^{pdb2}_{ideal}$	737.2
blind	572.9
$\mathbf{H}^{comax}_{mo}$	504.5

Coverage per heuristic, marginalised by planner.

The Association for the Advancement of Artificial Intelligence (AAAI 2023)