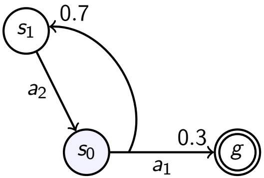
Heuristic Search for Multi-Objective Probabilistic Planning

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

AAAI 2023

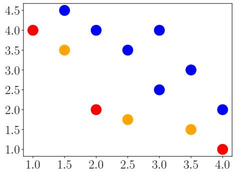
- stochastic shortest path problems (SSPs) are the defacto model for planning under uncertainty
  - compute a policy which maps states to actions
  - want optimal policy minimising the expected cost to reach the goal from an initial state



An SSP with a stochastic action a_1.

- heuristic search powerful for implicitly represented SSPs with large state spaces

- multi-objective stochastic shortest path problems (MOSSPs) generalise SSPs by exhibiting multiple objectives
  - aim to compute a set of non-dominated vector-valued policies with tradeoffs between objectives



orange+red=non-dominated vectors

- no existing heuristic search algorithms for MOSSPs
- algorithms that exist for MOMDPs do not carry over to MOSSPs: MOSSPs require additional assumptions

Contributions

- defining MOSSPs: a generalisation of MOMDPs
- assumptions for convergence of MOVI for MOSSPs
- heuristics for MOSSPs
- heuristic search algorithms for solving MOSSPs: iMOLAO*, MOLRTDP
- experimental evaluation

Contributions

defining MOSSPs: a generalisation of MOMDPs

assumptions for convergence of MOVI for MOSSPs

heuristics for MOSSPs

heuristic search algorithms for solving MOSSPs: iMOLAO*, MOLRTDP

experimental evaluation

Definition (MOSSPs)

An MOSSP is a tuple $P = (S, A, s_0, G)$

- S is a set of states
- A is a set of actions a which
  - maps certain states to a probability distribution of states, and
  - has an associated cost vector $\vec{C}(a) = [c_1, \ldots, c_n] \in \mathbb{R}^n_{\geq 0}$
- $s_0 \in S$ is an initial state, $G \subseteq S$ is a set of goal states

Informally,

A solution is a compact representation of a set of non-dominated vector-valued policies.

$\vec{u}$ *dominates* $\vec{v}$ denoted $\vec{u} \preceq \vec{v}$ iff $\vec{u}[i] \leq \vec{v}[i], i = 1, \ldots, n$

*Full definition: too many technical details and sub-definitions to fit into the talk

MO Bellman Backup

- fundamental equations for solving (MO)SSPs and (MO)MDPs
- states assigned $\mathbf{Q}$ and $\mathbf{V}$ values; each a finite set of vectors $\in \mathcal{P}_{<\omega}(\mathbb{R}^n_{\geq 0})$
- $\mathbf{Q} : S \times A \to \mathcal{P}_{<\omega}(\mathbb{R}^n_{\geq 0}) \simeq$ MO expected cost to goal when executing a at s
- $\mathbf{V} : S \to \mathcal{P}_{<\omega}(\mathbb{R}^n_{\geq 0}) \simeq$ MO expected cost to goal from $s \to$ *minimum* of $\mathbf{Q}$ values

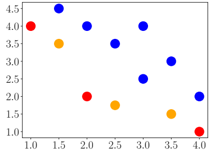
- For $s \in S \setminus G$:

$$\mathbf{V}^{t+1}(s) = \mathrm{CCS}\left(\bigcup_{a \in A} \mathbf{Q}^{t+1}(s,a)\right)$$

$$\mathbf{Q}^{t+1}(s,a) = \{\vec{C}(a)\} \oplus \left(\bigoplus_{s' \in S} P(s'|s,a)\mathbf{V}^t(s')\right)$$

- For $g \in G$:

$$\mathbf{V}^{t+1}(g) = \{\vec{0}\}$$



red: CCS;
orange+red: PCS

- CCS is the convex hull of a Pareto coverage set $\simeq$ MO generalisation of *min*
- $\mathbf{V} \oplus \mathbf{U} = \{\vec{u} + \vec{v} \mid \vec{u} \in \mathbf{U}, \vec{v} \in \mathbf{V}\}$ is setwise sum of vectors
- $\bigoplus$ generalises $\oplus$ to several sets

- $\lim_{t \to \infty} \mathbf{V}^t = \mathbf{V}^*$ [White, 1982; Barrett and Narayanan, 2008]

MO Value Iteration

Algorithm: MOVI

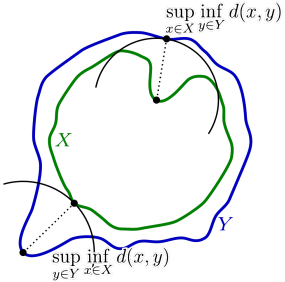
Data: MOSSP problem $P = (S, A, s_0, G)$, initial values $\mathbf{V}(s)$ for each state s (default to $\mathbf{V}(s) = \{\vec{0}\}$), and consistency threshold ε.

```
while max_{s∈S} res(s) < ε do
    for s ∈ S do
        if s ∈ G then  V_new(s) ← {0⃗} ;
        else  V_new(s) ←
            MOBellmanBackup(s) ;
        res(s) ← D(V, V_new)
    V ← V_new
return V
```

same as (single-objective) Value Iteration except

1. MO backups instead of SO backups
2. MO residual defined using Hausdorff metric (for finite sets of points):

$$D(\mathbf{U}, \mathbf{V}) = \max\left\{ \max_{\vec{u}\in\mathbf{U}} \min_{\vec{v}\in\mathbf{V}} d(\vec{u}, \vec{v}), \max_{\vec{u}\in\mathbf{V}} \min_{\vec{v}\in\mathbf{U}} d(\vec{u}, \vec{v}) \right\}$$



Contributions

defining MOSSPs: a generalisation of MOMDPs

assumptions for convergence of MOVI for MOSSPs

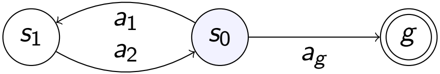
heuristics for MOSSPs

heuristic search algorithms for solving MOSSPs: iMOLAO*, MOLRTDP

experimental evaluation

Assumptions for convergence of MOVI for MOSSPs

Consider the MOSSP



MOSSP with action costs $\vec{C}(a_1) = [1, 0]$, $\vec{C}(a_2) = [1, 0]$, $\vec{C}(a_g) = [0, 1]$.

- $\pi_1 : S \to A, s_0 \mapsto a_g$ is a proper policy
- $\pi_2 : S \to A, s_0 \mapsto a_1, s_1 \mapsto a_2$ is an improper policy
- MOVI does not detect this; $\mathbf{V}(s_0) = \{[\infty, 0], [0, 1]\}$ in the limit
- VI able to detect and prune improper SSP policies which have ∞ cost

Assumptions for convergence of MOVI for MOSSPs

Strong improper policy assumption

- reachability assumption
  - $\forall s \in S$, $\exists$ proper policy at s
- all improper policies cost $\vec{\infty}$

- MOVI + strong assumption is sound and complete
- implies cycle costs have nonzero components (not easy to guarantee)
- one method: all action costs have nonzero components
  - cannot model independent costs
  - e.g. navigation domain: `load`, `unload` consumes 0 fuel, 1 time

Weak improper policy assumption

- reachability assumption
- exists a bound $\vec{b}$ s.t. $\forall s \in S$
  - for all proper π, $V^{\pi}(s) \preceq \vec{b}$
  - for all improper π, $V^{\pi}(s) \not\preceq \vec{b}$
- $\vec{u} \preceq \vec{v}$ iff $\vec{u}[i] \leq \vec{v}[i], i = 1, \ldots, n$

- *modified MOVI* + weak assumption is sound and complete
- can derive or estimate upper bound $\vec{b}$

Contributions

- defining MOSSPs: a generalisation of MOMDPs
- assumptions for convergence of MOVI for MOSSPs
- heuristics for MOSSPs
- heuristic search algorithms for solving MOSSPs: iMOLAO*, MOLRTDP
- experimental evaluation

Heuristic search

- heuristic search powerful for (optimal) planning
- does not require enumerating the whole potentially exponential search space
- heuristic search algorithms:

	SO	MO
determinstic	A^*, BiA^* etc.	$NAMOA^*$
stochastic	(i)LAO^*, LRTDP etc.	our contributions

MOSSP Heuristic

- admissible heuristics in single-objective (SO) deterministic case guarantee optimality with A*
- near optimality for SSPs with (i)LAO* and LRTDP in finitely many iterations
  - optimality with possibly infinitely many iterations
- in SO case, heuristic for a state is a scalar $h(s) \in \mathbb{R}_{\geq 0}$
- in MO case, heuristic for a state is a <u>set</u> of vectors $\mathbf{H}(s) \subset \mathbb{R}^n_{\geq 0}$

Definition (admissible MO heuristic)

$\mathbf{H}$ is *admissible* 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 $\forall g \in G, \mathbf{H}(g) = \{\vec{0}\}$.

- same definition as in deterministic MO case [Mandow and Pérez-de-la-Cruz, 2010]

Domain independent MOSSP Heuristics

- no heuristic: the zero heuristic defined by $\mathbf{H}_{zero}(s) = \{\vec{0}\}$
- can use SO heuristics
  - *ideal point heuristic*; apply an SO heuristic h_i to each objective in isolation:

$$\mathbf{H}_{ideal}(s) = \{[h_1(s), \dots h_n(s)]\}$$

  - e.g. $\mathbf{H}_{ideal}^{max}$, $\mathbf{H}_{ideal}^{pdb2}$, $\mathbf{H}_{ideal}^{pdb3}$ [Bonet and Geffner, 2001; Klößner and Hoffmann, 2021]
- can use deterministic MO heuristics
  - construct determinised problem by replacing each probabilistic effect with a deterministic action
  - e.g. $\mathbf{H}_{mo}^{comax}$, $\mathbf{H}_{mo}^{pdb2}$, $\mathbf{H}_{mo}^{pdb3}$ [Geißer et al., 2022]
- *new* abstraction heuristics
  - combine values from solving smaller projections of the problem
  - considers both MO and stoch. features of MOSSPs
  - e.g. $\mathbf{H}_{mossp}^{pdb2}$, $\mathbf{H}_{mossp}^{pdb3}$
- other existing SO and MO heuristics can be extended in above ways

Contributions

defining MOSSPs: a generalisation of MOMDPs

assumptions for convergence of MOVI for MOSSPs

heuristics for MOSSPs

heuristic search algorithms for solving MOSSPs: iMOLAO*, MOLRTDP

experimental evaluation

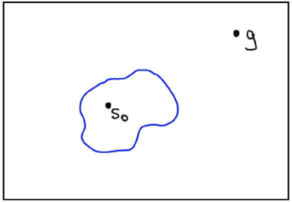
iMOLAO*

- gradually build partial solution via heuristic search

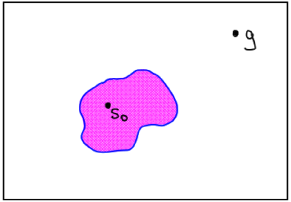
Algorithm: iMOLAO*

Data: MOSSP problem $P = (S, A, s_0, G)$, heuristic $\mathbf{H}$, and consistency threshold ε

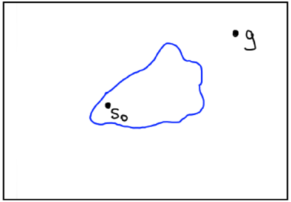
$\mathbf{V} \leftarrow \mathbf{H}$; $\Pi \leftarrow \emptyset$; $F \leftarrow \{s_0\}$; $I \leftarrow \emptyset$; $N \leftarrow \{s_0\}$
while $((F \cap N) \setminus G \neq \emptyset) \wedge (\max_{s \in N} res(s) < \varepsilon)$ **do**
    $F = \emptyset$
    $N \leftarrow \text{postorderTraversalDFS}(s_0, \Pi)$
    for $s \in N$ *in the computed order* **do**
        $\mathbf{V}(s) \leftarrow \mathrm{MOBellmanBackup}(s)$
        $\Pi(s) = \mathrm{getActions}(s, \mathbf{V})$
        if $s \notin I$ **then** $F = F \cup \{s\}$;
        $I = I \cup \{s\}$
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; then return to step 1 and repeat until covergence criteria is met

MOLRTDP

Algorithm: MOLRTDP

Data: MOSSP problem $P = (S, A, s_0, G)$, heuristic $\mathbf{H}$, and consistency threshold ε

```
procedure MOLRTDP(P, ε, H)
    V ← H
    while ¬s0.solved do
        visited ← ∅
        s ← s0
        while ¬s.solved do
            visited.push(s)
            if s ∈ G then break ;
            V(s) ← MOBellmanBackup(s)
            a ← sampleUnsolvedGreedyAction(s)
            s ← sampleUnsolvedNextState(s, a)
        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) ;
    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′) ;
    if rv then for s ∈ closed do s.solved = true ;
    else
        while closed ≠ ∅ do
            s ← closed.pop()
            V(s) ← MOBellmanBackup(s)
    return rv
```

MOLRTDP

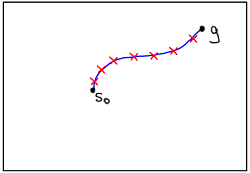
Algorithm: MOLRTDP

Data: MOSSP problem $P = (S, A, s_0, G)$, heuristic $\mathbf{H}$, consistency threshold ε

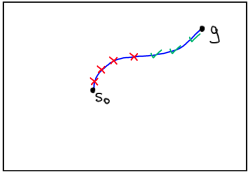
procedure $MOLRTDP(P, \varepsilon, \mathbf{H})$

- $\mathbf{V} \leftarrow \mathbf{H}$
- **while** $\neg s_0.solved$ **do**
  - $visited \leftarrow \emptyset$
  - $s \leftarrow s_0$
  - **while** $\neg s.solved$ **do**
    - $visited.push(s)$
    - **if** $s \in G$ **then** **break** ;
    - $\mathbf{V}(s) \leftarrow \mathrm{MOBellmanBackup}(s)$
    - $a \leftarrow \mathrm{sampleUnsolvedGreedyAction}(s)$
    - $s \leftarrow \mathrm{sampleUnsolvedNextState}(s, a)$
  - **while** $\neg visited.empty()$ **do**
    - $s \leftarrow visited.pop()$
    - **if** $\neg \mathrm{checkSolved}(s)$ **then** **break** ;
- **return** $\mathbf{V}$

- a state is 'solved' if its own and all its ancestors' values under the current partial solution have converged



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

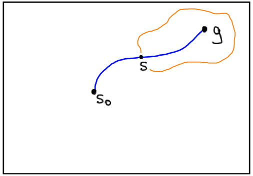


2. update whether states are solved in reverse order of the trialled walk using checkSolved; return to step 1 and repeat until s_0 is solved

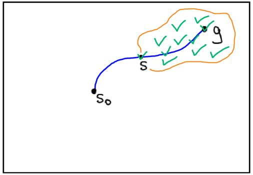
MOLRTDP - checkSolved

```
routine checkSolved(s)
  rv ← true; open ← ∅; closed ← ∅
  if ¬s.solved then  open.push(s) ;
  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′) ;

  if rv then  for s ∈ closed do
    s.solved = true ;
  else
    while closed ≠ ∅ do
      s ← closed.pop()
      V(s) ← MOBellmanBackup(s)

  return rv
```

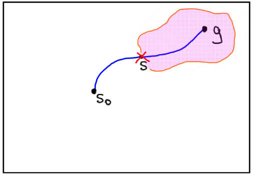
- a state is ‘solved’ if its own and all its ancestors’ values under the current partial solution have converged



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



2.2.a. if values for all ancestor states have converged, set the state and its ancestors to be solved



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

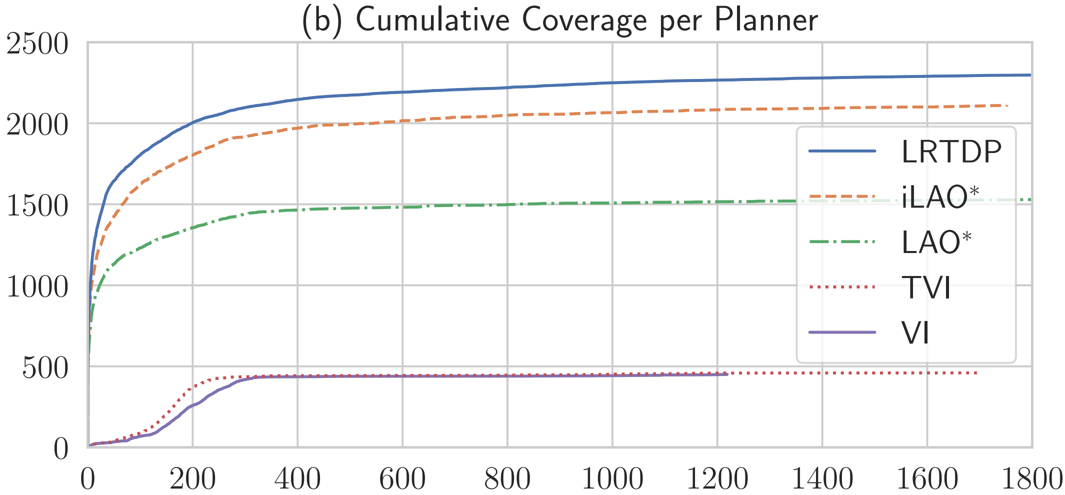
Contributions

- defining MOSSPs: a generalisation of MOMDPs
- assumptions for convergence of MOVI for MOSSPs
- heuristics for MOSSPs
- heuristic search algorithms for solving MOSSPs: iMOLAO*, MOLRTDP
- experimental evaluation

Experimental setup

- 5 solvers (MO prefix omitted): VI, TVI, LAO*, iLAO*, LRTDP
- 9 heuristics: zero, $\mathbf{H}^{max}_{ideal}$, $\mathbf{H}^{comax}_{mo}$, $\mathbf{H}^{pdb2}_{ideal}$, $\mathbf{H}^{pdb3}_{ideal}$, $\mathbf{H}^{pdb2}_{mo}$, $\mathbf{H}^{pdb3}_{mo}$, $\mathbf{H}^{pdb2}_{mossp}$, $\mathbf{H}^{pdb3}_{mossp}$
- 30 minute timeout, single CPU core, 4GB memory
- 7 domains, mix of MO and probabilistic interesting domains
- 610 total problems
- $5 \times 9 \times 610 = 27450$ possible experimental configurations

Best planner



Cumulative coverage per planner, marginalised by heuristics.

Best heuristic

Heuristic	Normalised coverage
$\mathbf{H}^{pdb3}_{mossp}$	19.1
$\mathbf{H}^{pdb2}_{mossp}$	17.9
$\mathbf{H}^{pdb3}_{mo}$	17.4
$\mathbf{H}^{pdb2}_{mo}$	17.1
$\mathbf{H}^{max}_{ideal}$	14.9
$\mathbf{H}^{pdb3}_{ideal}$	14.7
$\mathbf{H}^{pdb2}_{ideal}$	14.4
blind	12.1
$\mathbf{H}^{comax}_{mo}$	10.7

Heuristic	Unnormalised 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

- marginalise by planner
- normalisation is done over domain due to uneven number of problems
- ranking of top 3 and bottom 3 heuristics same

Heuristic accuracy

		Critical Path		Abstractions						
	blind	$\mathbf{H}^{max}_{ideal}$	$\mathbf{H}^{comax}_{mo}$	$\mathbf{H}^{pdb2}_{ideal}$	$\mathbf{H}^{pdb3}_{ideal}$	$\mathbf{H}^{pdb2}_{mo}$	$\mathbf{H}^{pdb3}_{mo}$	$\mathbf{H}^{pdb2}_{mossp}$	$\mathbf{H}^{pdb3}_{mossp}$
SAR-4	100	97	44	93	92	44	44	38	**26**
SAR-5	100	97	45	93	92	45	45	39	**28**
ExBw-2d	100	59	**24**	52	45	52	45	52	45
ExBw-3d	100	58	**22**	52	44	52	44	52	44
Tireworld	100	100	68	100	100	68	68	57	**12**
VisitAll	100	100	54	100	100	61	53	22	**12**
VisitAllTire	100	100	50	100	100	66	**45**	66	**45**

- mean relative error (%) of heuristic value at the initial state relative to the optimal value for solved instances
- error calculated by directed Hausdorff metric difference divided by the norm of the largest vector of the optimal value:

$$\max_{\vec{v}\in\mathbf{V}} \min_{\vec{u}\in\mathbf{H}} d(\vec{v},\vec{u}) / \max_{\vec{v}\in\mathbf{V}} \|\vec{v}\|$$

$\Longrightarrow$ MO features more important than stoch. features to capture in a heuristic

Thank you for your attention!