Markovian State and Action Abstractions for MDPs via Hierarchical MCTS

Aijun Bai, Siddharth Srivastava, Stuart Russell
aijunbai@berkeley.edu, srivass@utrc.utc.com, russell@cs.berkeley.edu

MDPs and POMDPs

A Markov decision process (MDP) is a tuple $\langle S, A, T, R, \gamma \rangle$:

- State space: S
- Action space: A
- Transition function: $T(s' \mid s, a)$
- Reward function: $R(s, a)$
- Discount factor: γ

A partially observable Markov decision process (POMDP) is a tuple $\langle S, A, Z, T, R, \Omega, \gamma \rangle$:

- Underlying MDP: $\langle S, A, T, R, \gamma \rangle$
- Observation space: Z
- Observation function: $\Omega(z \mid s, a)$

State and Action Abstractions

State abstraction groups a set of states as a unit:

- Ground MDP: $M = \langle S, A, T, R, \gamma \rangle$
- Partition: $X = \{x_1, x_2, \cdots\}$
- Abstraction function: $\varphi : S \to X$
- Non-Markovianess: $\Pr(x' \mid x, a)$
- Non-Stationary: $\Pr(s \mid x)$

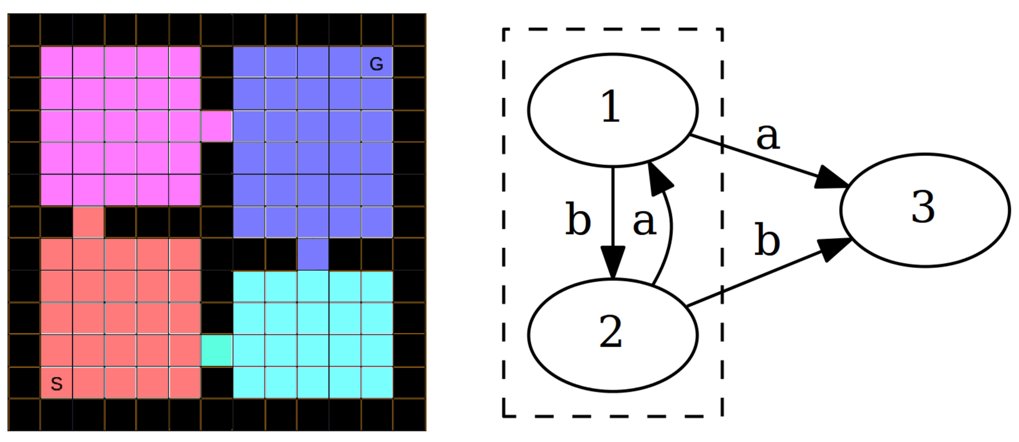

Figure 1: Rooms domain (L); Non-Markovianess (R)

Action abstraction extends the macro idea to closed-loop policies:

- Option o: initial and terminal conditions and an inner policy π_o
- Semi MDP: $\langle S, \mathcal{O}, T_{\mathcal{O}}, R, \gamma \rangle$
- Temporal transition: $T_{\mathcal{O}}(s', N \mid s, o)$

Motivation

The *safe state abstraction* approach:

- Ignore only irrelevant state variables
- Exploit *bisimulation/homomorphism*
- Not always possible
- Computationally difficult to find

The *weighting function* approach:

- Approximate $\Pr(s \mid x)$ using $w(s, x)$
- Superficially ensure Markovianess
- Can not capture the true dynamics
- $\Pr(s \mid x)$ is non-stationary anyway

State Abstraction as a POMDP

State abstraction creates a POMDP:

- Ground MDP: $M = \langle S, A, T, R, \gamma \rangle$
- Partial observability: X as observations
- Obs. function: $\Omega(x \mid s) = \mathbf{1}[x = \varphi(s)]$
- POMDP$(M, \varphi) = \langle S, A, X, T, R, \Omega, \gamma \rangle$
- M-branching factor: $|S| \times |A|$
- POMDP(M, φ)-branching factor: $|X| \times |A|$
- POMCP(M, φ): POMCP on the POMDP

Action Abstraction with Belief

A given state abstraction naturally induces an action abstraction:

- Options connect abstract states in a one high-level step
- E.g., transition from x to y as option $o_{x \to y}$

Hierarchical solution for POMDP(M, φ) with $\mathcal{O}$ as the set of options:

- The overall option-selection policy: μ
- Inner policy π_o for option $o \in \mathcal{O}$
- A set of policies $\Pi = \{\mu, \pi_{o_1}, \pi_{o_2}, \cdots\}$

Value function decomposition:

- $V^{\mu}(h) = \max_o Q^{\mu}(h, o)$
- $Q^{\mu}(h, o) = V^{\pi_o}(h) + \sum_{h' \in \mathcal{H}} \gamma^{|h'| - |h|} \Pr(h' \mid h, o) V^{\mu}(h')$
- $V^{\pi_o}(h) = \max_a Q^{\pi_o}(h, a)$
- $Q^{\pi_o}(h, a) = R(h, a) + \gamma \sum_{x \in X} \Pr(x \mid h, a) V^{\pi_o}(hax)$
- POMCP$(M, \varphi, \mathcal{O})$: hierarchical POMCP

Theoretical Results

Theorem 1 *In the limit, POMCP(M, φ) finds the optimal policy for ground MDP M consistent with input state abstraction φ.*

Definition 1 *An aggregation error is defined to measure the quality of state abstraction in terms of grouping states with different optimal actions.*

Theorem 2 *The performance loss in terms of action values of POMCP(M, φ) is bounded by a constant multiple of the aggregation error.*

Theorem 3 *In the limit, POMCP$(M, \varphi, \mathcal{O})$ converges to a recursively optimal hierarchical policy for POMDP(M, φ) over the hierarchy defined by state abstraction φ and options $\mathcal{O}$.*

References

Please refer to the original paper for more details, also available at the author's personal homepage: `http://aijunbai.net/`.

Experimental Results

We compare 5 algorithms in ROOMS$[17, 17, 4]$ and ROOMS$[25, 13, 8]$ problems: UCT, UCT$_{\varphi}$, POMCP(M, φ), POMCP$(M, \varphi, \mathcal{O})$ and smart-POMCP$(M, \varphi, \mathcal{O})$. UCT runs directly in the ground state space; UCT$_{\varphi}$ is a UCT algorithm running entirely in the abstract state space following the weighting function approach; POMCP(M, φ) is a POMCP algorithm running on POMDP(M, φ); POMCP$(M, \varphi, \mathcal{O})$ is the proposed hierarchical MCTS algorithm running on POMDP(M, φ); smart-POMCP$(M, \varphi, \mathcal{O})$ is a POMCP$(M, \varphi, \mathcal{O})$ algorithm equipped with hand-coded informative rollout policies for options. The performance is evaluated using averaged discounted return in terms of the number of simulations and the averaged computation time per action.

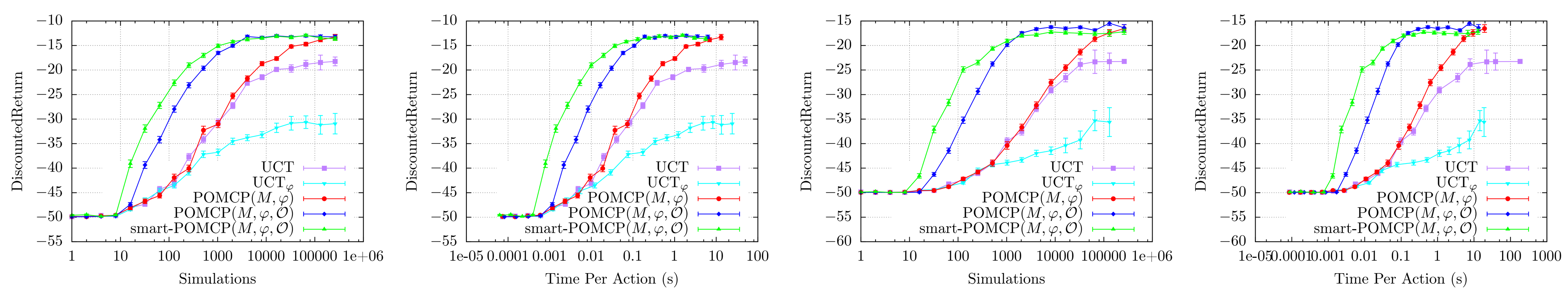

Figure 2: Experimental results on the rooms domain: ROOMS$[17, 17, 4]$ (L); ROOMS$[25, 13, 8]$ (R).

POMCP(M, φ) outperforms UCT, indicating that modeling a ground MDP with state abstraction as a POMDP and solving it via approximated, search-based online planning algorithms is feasible. UCT$_{\varphi}$ uses the empirical distributions of $\Pr(s \mid x)$ to approximate $w(s, x)$ and finds a memoryless policy as a mapping from abstract states to actions. It has easily the worst performance in all cases, confirming that finding memoryless policies might not be the right way to do state abstractions. POMCP$(M, \varphi, \mathcal{O})$ outperforms UCT by orders of magnitude. POMCP$(M, \varphi, \mathcal{O})$ also outperforms POMCP(M, φ) substantially suggesting that exploiting the hierarchical structure introduced by doing state abstraction contributes the main improvement. With the help of an option-specific rollout policy, smart-POMCP$(M, \varphi, \mathcal{O})$ improves on POMCP$(M, \varphi, \mathcal{O})$ significantly, indicating the advantage that option-specific heuristic can be added to POMCP$(M, \varphi, \mathcal{O})$.

Conclusions

- We propose state- and action-abstracted MDPs can be viewed as POMDPs
- We bound the performance loss induced by the abstraction
- We describe a hierarchical MCTS algorithm for approximately solving the abstract POMDP
- The algorithm converges to a recursively optimal hierarchical policy for the ground MDP consistent with the input state and action abstractions
- Empirical results show that the proposed approach improves ground MCTS significantly

Future Work

- Reinforcement learning with features creates a POMDP
- Memoryless policies in feature space lack optimality guarantee
- The non-Markovianess can be overcome by using a POMDP formulation
- The hierarchical structure in feature space can be exploited by hierarchical MCTS