Finding Options that Minimize Planning Time

Yuu Jinnai [1] David Abel [1] D Ellis Hershkowitz [2] Michael L. Littman [1] George Konidaris [1]

Abstract

We formalize the problem of selecting the optimal set of options for planning as that of computing the smallest set of options so that planning converges in less than a given maximum of value-iteration passes. We first show that the problem is *NP*-hard, even if the task is constrained to be deterministic—the first such complexity result for option discovery. We then present the first polynomial-time boundedly suboptimal approximation algorithm for this setting, and empirically evaluate it against both the optimal options and a representative collection of heuristic approaches in simple grid-based domains.

1. Introduction

Markov Decision Processes or MDPs (Puterman, 1994) are a widely used expressive model of sequential decision-making. However, MDPs are computationally expensive to solve (Papadimitriou & Tsitsiklis, 1987; Littman, 1997; Goldsmith et al., 1997). One approach to solving such problems is to add high-level, temporally extended actions—often formalized as options (Sutton et al., 1999)—to the set of actions available to the agent. The right set of options allows planning to probe more deeply into the search space with a single computation. Thus, if options are chosen appropriately, planning algorithms can find good plans with less computation.

Indeed, previous work has offered substantial support that abstract actions can accelerate planning (Mann & Mannor, 2014; Silver & Ciosek, 2012). However, little is known about how to find the right set of options for planning. Prior work often seeks to codify an intuitive notion of what underlies an effective option, such as identifying relatively novel states (Şimşek & Barto, 2004), identifying bottleneck states or high-betweenness states (Şimşek et al., 2005; Şimşek & Barto, 2009; Bacon, 2013; Moradi et al., 2012), finding repeated policy fragments (Pickett & Barto, 2002), or finding states that often occur on successful trajectories (McGovern & Barto, 2001; Bakker & Schmidhuber, 2004). While such intuitions often capture important aspects of the role of options in planning, the resulting algorithms are somewhat heuristic in that they are not based on optimizing any precise performance-related metric; consequently, their relative performance can only be evaluated empirically.

We aim to formalize what it means to find the set of options that is optimal for planning, and to use the resulting formalization to develop an algorithm with performance guarantees and a principled theoretical foundation. Specifically, we consider the problem of finding the smallest set of options so that planning converges in fewer than ℓ value iterations (VI). Our main result is that this problem is *NP*-hard. More precisely, the problem:

1. is $2^{\log^{1-\epsilon} n}$-hard to approximate for any $\epsilon > 0$ unless $NP \subseteq DTIME(n^{\text{poly}\log n})$,[1] where n is the input size;
2. is $\Omega(\log n)$-hard to approximate even for deterministic MDPs unless $P = NP$;
3. has an $O(n)$-approximation algorithm;
4. has an $O(\log n)$-approximation algorithm for deterministic MDPs.

In Section 4, we introduce A-MOMI, a polynomial-time approximation algorithm that has $O(n)$ suboptimality in general and $O(\log n)$ suboptimality for deterministic MDPs. The expression $2^{\log^{1-\epsilon} n}$ is only slightly smaller than n: if $\epsilon = 0$ then $\Omega(2^{\log n}) = \Omega(n)$. Thus, the inapproximability results claim that A-MOMI is close to the best possible approximation factor.

In addition, we consider the complementary problem of finding a set of k options that minimize the number of VI iterations until convergence. We show that this problem is also *NP*-hard, even for a deterministic MDP and introduce A-MIMO, a polynomial time approximation algorithm.

[1]Brown University, Providence, RI, United States [2]Carnegie Mellon University, Pittsburgh, PA, United States. Correspondence to: Yuu Jinnai <yuu_jinnai@brown.edu>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

[1]This is a standard complexity assumption: See, for example, Dinitz et al. (2012).

Finally, we empirically evaluate the performance of two heuristic approaches for option discovery, betweenness options (Şimşek & Barto, 2009) and Eigenoptions (Machado et al., 2017), against the proposed approximation algorithms and the optimal options in standard grid domains.

2. Background

We first provide background on Markov Decision Processes (MDPs), planning, and options.

2.1. Markov Decision Processes and Planning

An MDP is a five tuple: $\langle \mathcal{S}, \mathcal{A}, R, T, \gamma \rangle$, where $\mathcal{S}$ is a finite set of states; $\mathcal{A}$ is a finite set of actions; $R : \mathcal{S} \times \mathcal{A} \rightarrow [0, \text{RMAX}]$ is a reward function; $T : \mathcal{S} \times \mathcal{A} \rightarrow \Pr(\mathcal{S})$ is a transition function, denoting the probability of arriving in state $s' \in \mathcal{S}$ after executing action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$; and $\gamma \in [0, 1]$ is a discount factor, expressing the agent's preference for immediate over delayed rewards.

An action-selection strategy is modeled by a *policy*, $\pi : \mathcal{S} \rightarrow \Pr(\mathcal{A})$, mapping states to a distribution over actions. Typically, the goal of planning in an MDP is to *solve* the MDP—that is, to compute an optimal policy. A policy π is evaluated according to the Bellman equation, denoting the long term expected reward received by executing π:

$$V^\pi(s) = R(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} T(s, \pi(s), s') V^\pi(s'). \quad (1)$$

We denote $\pi^*(s) = \arg\max_\pi V^\pi(s)$ and $V^*(s) = \max_\pi V^\pi(s)$ as the optimal policy and value function, respectively.

The core problem we study is *planning*, namely, computing a near optimal policy for a given MDP. The main variant of the planning problem we study is the *value-planning problem*:

Definition 1 (Value-Planning Problem): **Given** *an MDP* $M = \langle \mathcal{S}, \mathcal{A}, R, T, \gamma \rangle$ *and a non-negative real-value* ϵ, **return** *a value function,* V *such that* $|V(s) - V^*(s)| < \epsilon$ *for all* $s \in \mathcal{S}$.

The value-planning problem can be solved in time polynomial in the size of the state space (Littman et al., 1995).

2.2. Options and Value Iteration

Temporally extended actions offer great potential for mitigating the difficulty of solving complex MDPs, either through planning or reinforcement learning (Sutton et al., 1999). However, it is possible that options that are useful for learning are not necessarily useful for planning, and vice versa. In fact, we don't have an explicit metric for measuring the quality of an option set for planning. Therefore, identifying techniques that produce good options in these scenarios is an important open problem in the literature.

We use the standard definition of options (Sutton et al., 1999):

Definition 2 (option): *An option* o *is defined by a triple:* $(\mathcal{I}, \pi, \beta)$ *where:*

- $\mathcal{I} \subseteq \mathcal{S}$ *is a set of states where the option can initiate,*
- $\pi : \mathcal{S} \rightarrow \Pr(\mathcal{A})$ *is a policy,*
- $\beta : \mathcal{S} \rightarrow [0, 1]$, *is a termination condition.*

We let $\mathcal{O}_{all}$ *denote the set containing all options.*

In planning, options have a well defined transition and reward model for each state named the multi-time model, introduced by Precup & Sutton (1998):

$$T_\gamma(s, o, s') = \sum_{t=0}^{\infty} \gamma^t \Pr(s_t = s', \beta(s_t) \mid s, o). \quad (2)$$

$$R_\gamma(s, o) = \mathop{\mathbb{E}}_{o_\pi} \left[r_1 + \gamma r_2 + \ldots + \gamma^{k-1} r_k \;\middle|\; s, o \right]. \quad (3)$$

We use the multi-time model for value iteration. The algorithm computes a sequence of functions $V_0, V_1, ..., V_b$ using the Bellman optimality operator on the multi-time model:

$$V_{i+1}(s) = \max_{o \in A \cup \mathcal{O}} \left(R_\gamma(s, o) + \sum_{s' \in S} T_\gamma(s, o, s') V_i(s') \right). \quad (4)$$

The problem we consider is to find a set of options to add to the set of primitive actions that minimize the number of iterations required for VI to converge:[2]

Definition 3 ($L_{\epsilon, V_0}(\mathcal{O})$): *The number of iterations* $L_{\epsilon, V_0}(\mathcal{O})$ *of VI using the joint action set* $\mathcal{A} \cup \mathcal{O}$*, with* $\mathcal{O}$ *a non-empty set of options, is the smallest* b *at which* $|V_{b'}(s) - V^*(s)| < \epsilon$ *for all* $s \in \mathcal{S}$, $b' \geq b$.

2.2.1. Point Options

The options formalism is immensely general. Due to its generality, a single option can actually encode several completely unrelated sets of different behaviors. Consider the nine-state example MDP pictured in Figure 1; a single option can in fact initiate, make decisions in, and terminate along entirely independent trajectories. As we consider more complex MDPs (which, as discussed earlier, is often a motivation for introducing options), the number of inde-

[2]We can ensure $|V^*(s) - V_i(s)| < \epsilon$ by running VI until $|V_{i+1}(s) - V_i(s)| < \epsilon(1 - \gamma)/2\gamma$ for all $s \in \mathcal{S}$ (Williams & Baird, 1993).

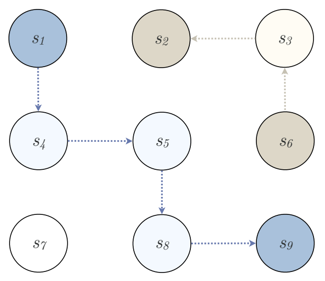


Figure 1: A single option can encode multiple unrelated behaviors. The dark circles indicate where the option can be initiated (s_1 & s_6) and terminated (s_2 & s_9), whereas the lighter circles denote the states visited by the option policy when applied in the respective initiating state.

pendent behaviors that can be encoded by a single option increases further still.

As a result, it can be difficult to reason about the impact of adding a single option, in the traditional sense. As the MDP grows larger, a combinatorial number of different behaviors can emerge from "one" option. Consequently, it is difficult to address the question: which *single* option helps planning the most? As MDPs grow large, one option can encode a large number of possible, independent behaviors. Thus, we instead introduce and study "point options", which only allow for a single continuous stream of behavior:

Definition 4 (Point option): *A* **point option** *is any option whose initiation set and termination set are each true for exactly one state each:*

$$|\{s \in \mathcal{S} : \mathcal{I}(s) = 1\}| = 1, \quad (5)$$
$$|\{s \in \mathcal{S} : \beta(s) > 0\}| = 1, \quad (6)$$
$$|\{s \in \mathcal{S} : \beta(s) = 1\}| = 1. \quad (7)$$

We let $\mathcal{O}_p$ denote the set containing all point options.

For simplicity, we denote the initiation state as $\mathcal{I}_o$ and the termination state as β_o for a point option o.

To plan with a point option from state s, the agent runs value iteration using a model (Eq. 2, 3) in addition to the backup operations by primitive actions where k is the duration of the option. We assume that the model of each option is given to the agent and ignore the computation cost for computing the model for the options.

Point options are a useful subclass to consider for several reasons. First, a point option is a simple model for a temporally extended action. Second, the policy of the point option can be calculated as a path-planning problem for deterministic MDPs. Third, any other options with a single termination state with termination probability 1 can be represented as a collection of point options. Fourth, a point option has constant computational overhead per iteration.

3. Complexity Results

Our main results focus on two computational problems:

1. MINOPTIONMAXITER (MOMI): Which set of options lets value iteration converge in at most ℓ iterations?
2. MINITERMAXOPTION (MIMO): Which set of k or fewer options minimizes the number of iterations to convergence?

More formally, MOMI is defined as follows.

Definition 5 (MOMI): *The* MINOPTIONMAXITER *problem:*
Given *an MDP M, a non-negative real-value ϵ, an initial value function V_0, and an integer ℓ* **return** *$\mathcal{O}$ that minimizes $|\mathcal{O}|$ subject to $\mathcal{O} \subseteq \mathcal{O}_p$ and $L_{\epsilon,V_0}(\mathcal{O}) \leq \ell$.*

We then consider the complementary optimization problem MINITERMAXOPTION (MIMO): compute a set of k options which minimizes the number of iterations:

Definition 6 (MIMO): *The* MINITERMAXOPTION *problem:*
Given *an MDP M, a non-negative real-value ϵ, an initial value function V_0, and an integer k* **return** *$\mathcal{O}$ that minimizes $L_{\epsilon,V_0}(\mathcal{O})$, subject to $\mathcal{O} \subseteq \mathcal{O}_p$ and $|\mathcal{O}| \leq k$.*

We now introduce our main result, which shows that both MOMI and MIMO are *NP*-hard.

Theorem 1. *MOMI and MIMO are NP-hard.*

Proof. We consider a problem OI-DEC which is a decision version of MOMI and MIMO. The problem asks if we can solve the MDP within ℓ iterations using at most k point options.

Definition 7 (OI-DEC):
Given *an MDP M, a non-negative real-value ϵ, an initial value function V_0, and integers k and ℓ,* **return** *'Yes' if the there exists an option set $\mathcal{O}$ such that $\mathcal{O} \subseteq \mathcal{O}_p$, $|\mathcal{O}| \leq k$ and $L_{\epsilon,V_0}(\mathcal{O}) \leq \ell$. 'No' otherwise.*

We prove the theorem by reduction from the decision version of the set-cover problem—known to be NP-complete—to OI-DEC. The set-cover problem is defined as follows.

Definition 8 (SetCover-DEC):
Given *a set of elements $\mathcal{U}$, a set of subsets $\mathcal{X} = \{X \subseteq \mathcal{U}\}$, and an integer k,* **return** *'Yes' if there exists a*

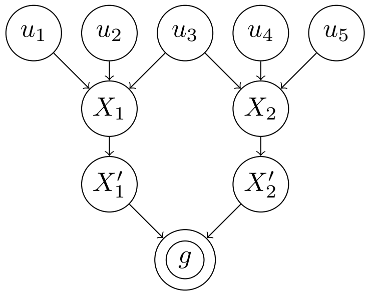


Figure 2: Reduction from SetCover-DEC to OI-DEC. The example shows the reduction from an instance of SetCover-DEC which asks if we can pick two subsets from $\mathcal{X} = \{X_1, X_2\}$ where $X_1 = \{1, 2, 3\}, X_2 = \{3, 4, 5\}$ to cover all elements $\mathcal{U} = \{1, 2, 3, 4, 5\}$. The SetCover-DEC can be reduced to an instance of OI-DEC where the question is whether the MDP can be solved with 2 iterations of VI by adding at most two point options. The answer of OI-DEC is 'Yes' (adding point options from X_1 and X_2 to g will solve the problem), thus the answer of the SetCover-DEC is 'Yes'. Here the set of initial states corresponds to the cover for the SetCover-DEC.

cover $\mathcal{C} \subseteq \mathcal{X}$ such that $\bigcup_{X \in \mathcal{C}} X = \mathcal{U}$ and $|\mathcal{C}| \leq k$. 'No' otherwise.

If there is some $u \in \mathcal{U}$ that is not included in at least one of the subsets X, then the answer is 'No'. Assuming otherwise, we construct an instance of a shortest path problem (a special case of an MDP problem) as follows (Figure 2). There are four types of states in the MDP: (1) $u_i \in \mathcal{U}$ represents one of the elements in $\mathcal{U}$, (2) $X_i \in \mathcal{X}$ represents one of the subsets in $\mathcal{X}$, (3) $X_i' \in \mathcal{X}'$: we make a copy for every state $X_i \in \mathcal{X}$ and call them X_i', (4) a goal state g. Thus, the state set is $\mathcal{U} \cup \mathcal{X} \cup \mathcal{X}' \cup \{g\}$. We build edges between states as follows: (1) $e(u, X) \in E$ iff $u \in X$: For $u \in \mathcal{U}$ and $X \in \mathcal{X}$, there is an edge between u and X. (2) $\forall X_i \in \mathcal{X}$, $e(X_i, X_i') \in E$: For every $X_i \in \mathcal{X}$, we have an edge from X_i to X_i'. (3) $\forall e(X', g) \in E$: for every $X' \in \mathcal{X}_i'$ we have an edge from X_i to the goal g. This construction can be done in polynomial time.

Let M be the MDP constructed in this way. We show that SetCover($\mathcal{U}, \mathcal{X}, k$) = OI-DEC($M, V_0 = 0, k, 2$). Note that by construction every state X_i, X_i', and g converges to its optimal value within 2 iterations as it reaches the goal state g within 2 steps. A state $u \in \mathcal{U}$ converges within 2 steps if and only if there exists a point option (a) from X to g where $u \in X$, (b) from u to X' where $u \in X$, or (c) from u to g. For options of type (b) and (c), we can find an option of type (a) that makes u converge within 2 steps by setting the initial state of the option to $\mathcal{I}_o = X$, where $u \in X$, and the termination state to $\beta_o = g$. Let $\mathcal{O}$ be the solution of OI-DEC($M, k, 2$). If there exists an option of type (b) or (c), we can swap them with an option of type (a) and still maintain a solution. Thus, it is sufficient to consifer that every option is type (a). Let $\mathcal{C}$ be a set of initial states of each option in $\mathcal{O}$ ($\mathcal{C} = \{\mathcal{I}_o | o \in \mathcal{O}\}$). This construction exactly matches the solution of the SetCover-DEC. □

3.1. Generalizations of MOMI and MIMO

A natural question is whether Theorem 1 extends to more general option-construction settings. We consider two possible extensions, which we believe offer significant coverage of finding optimal options for planning in general.

We first consider the case where the options are not necessarily point options. There is little sense in considering MOMI where one can choose any option since clearly the best option is the option whose policy is the optimal policy. Thus, using the space of all options $\mathcal{O}_{all}$ we generalize MOMI as follows (MIMO$_{gen}$ are defined analogously):

Definition 9 (MOMI$_{gen}$):
Given *an MDP M, a non-negative real-value ϵ, an initial value function V_0, $\mathcal{O}' \subseteq \mathcal{O}_{all}$, and an integer ℓ,* **return** *$\mathcal{O}$ minimizing $|\mathcal{O}|$ subject to $L_{\epsilon,V_0}(\mathcal{O}) \leq \ell$ and $\mathcal{O} \subseteq \mathcal{O}'$.*

Theorem 2. *MOMI$_{gen}$ and MIMO$_{gen}$ are NP-hard.*

The proof follows from the fact that MOMI$_{gen}$ is a superset of MOMI and MIMO$_{gen}$ is a superset of MIMO.

We next consider the multi-task generalization, where we aim to find a smallest number of options which the expected number of iterations to solve a problem M sampled from a distribution of MDPs, D, is bounded:

Definition 10 (MOMI$_{multi}$):
Given *A distribution of MDPs D, $\mathcal{O}' \subseteq \mathcal{O}_{all}$, a non-negative real-value ϵ, an initial value function V_0, and an integer ℓ,* **return** *$\mathcal{O}$ that minimizes $|\mathcal{O}|$ such that $E_{M \sim D}[L_M(\mathcal{O})] \leq \ell$ and $\mathcal{O} \subseteq \mathcal{O}'$.*

Theorem 3. *MOMI$_{multi}$ and MIMO$_{multi}$ are NP-hard.*

The proof follows from the fact that MOMI$_{multi}$ is a superset of MOMI$_{gen}$ and MIMO$_{multi}$ is a superset of MIMO$_{gen}$.

In light of the computational difficulty of both problems, the appropriate approach is to find tractable approximation algorithms. However, even approximately solving MOMI is hard. More precisely:

Theorem 4.

1. *MOMI is $\Omega(\log n)$ hard to approximate even for deterministic MDPs unless $P = NP$.*
2. *MOMI$_{gen}$ is $2^{\log^{1-\epsilon} n}$-hard to approximate for any $\epsilon > 0$ even for deterministic MDPs unless NP $\subseteq$ $DTIME(n^{poly \log n})$.*

3. *MOMI is $2^{\log^{1-\epsilon} n}$-hard to approximate for any $\epsilon > 0$ unless NP $\subseteq DTIME(n^{poly \log n})$.*

Proof. See appendix. □

Note that an $O(n)$-approximation is achievable by the trivial algorithm that returns a set of all candidate options. Thus, Theorem 4 roughly states that there is no polynomial time approximation algorithms other than the trivial algorithm for MOMI.

In the next section we show that an $O(\log n)$-approximation is achievable for MOMI if the MDP is deterministic and the agent is given a set of all point options. Thus, together, these two results give a formal separation between the hardness of abstraction in MDPs with and without stochasticity.

In summary, *the problem of computing optimal behavioral abstractions for planning is computationally intractable*.

4. Approximation Algorithms

We now provide polynomial-time approximation algorithms, A-MIMO and A-MOMI, to solve MOMI and MIMO, respectively. Both algorithms have bounded suboptimality slightly worse than a constant factor for deterministic MDPs.

We assume that (1) there is exactly one absorbing state $g \in \mathcal{S}$ with $T(g, a, g) = 1$ and $R(g, a) = 0$, and every optimal policy eventually reaches g with probability 1, (2) there is no cycle with a positive reward involved in the optimal policy's trajectory. That is, $V_+^\pi(s) := \mathbb{E}[\sum_{t=0}^\infty \max\{0, R(s_t, a_t)\}] < \infty$ for all policies π. Note that we can convert a problem with multiple goals to a problem with a single goal by adding a new absorbing state g to the MDP and adding a transition from each of the original goals to g.

Unfortunately, these algorithms are computationally harder than solving the MDP itself, and are thus not practical for planning. Instead, they are useful for analyzing and evaluating heuristically generated options. If the option set generated by the heuristic methods outperforms the option set found by the following algorithms, then one can claim that the option set found by the heuristic is close to the optimal option set (for that MDP). Our algorithms have a formal guarantee on bounded suboptimality if the MDP is deterministic, so any heuristic method that provably exceeds our algorithm's performance will also guarantee bounded suboptimality. We also believe these algorithms may be a useful foundation for future option discovery methods.

4.1. A-MOMI

We now describe a polynomial-time approximation algorithm, A-MOMI, based on using set cover to solve MOMI. The overview of the procedure is as follows.

1. Compute an asymmetric distance function $d_\epsilon(s, s') : S \times S \rightarrow \mathbb{N}$ representing the number of iterations for a state s to reach its ϵ-optimal value if we add a point option from a state s' to a goal state g.
2. For every state s_i, compute a set of states X_{s_i} within $\ell - 1$ distance of reaching s_i. The set X_{s_i} represents the states that converge within ℓ steps if we add a point option from s_i to g.
3. Let $\mathcal{X}$ be a set of X_{s_i} for every $s_i \in \mathcal{S} \setminus X_g^+$, where X_g^+ is a set of states that converges within ℓ without any options (thus can be ignored).
4. Solve the set-cover optimization problem to find a set of subsets that covers the entire state set using the approximation algorithm by Chvatal (1979). This process corresponds to finding a minimum set of subsets $\{X_{s_i}\}$ that makes every state in $\mathcal{S}$ converge within ℓ steps.
5. Generate a set of point options with initiation states set to one of the center states in the solution of the set-cover, and termination states set to the goal.

We compute a distance function $d_\epsilon : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{N}^3$, defined as follows:

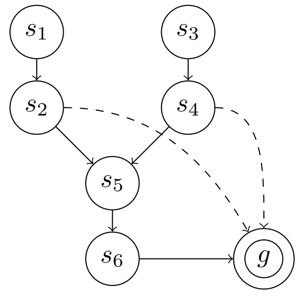
Definition 11 (Distance $d_\epsilon(s_i, s_j)$): *$d_\epsilon(s_i, s_j)$ is the smallest number b such that for all $b' \geq b$, $V'_b(s_i)$ is ϵ-optimal if we add a point option from s_j to g, minus one.*

More formally, let $d'_\epsilon(s_i)$ denote the number of iterations needed for the value of state s_i to satisfy $|V(s_i) - V^*(s_i)| < \epsilon$, and let $d'_\epsilon(s_i, s_j)$ be an upper bound of the number of iterations needed for the value of s_i to satisfy $|V(s_i) - V^*(s_i)| < \epsilon$, if the value of s_j is initialized such that $|V(s_j) - V^*(s_j)| < \epsilon$. We define $d_\epsilon(s_i, s_j) := \min(d'_\epsilon(s_i) - 1, d'_\epsilon(s_i, s_j))$. For simplicity, we use d to denote the function d_ϵ. Consider the following example.

Example. Table 3b is a distance function for the MDP shown in Figure 3a. For a deterministic MDP, $d_0(s)$ corresponds to the number of edge traversals from state s to g, where we have edges only for those that corresponds to the state transition by the optimal actions. The quantity $d_0(s, s') - 1$ is the minimum of $d_0(s)$ and one plus the number of edge traversals from s to s'. ◇

Note that we only need to solve the MDP once to compute d. $d(s, s')$ can be computed once you solved the MDP without any options and store all value functions V_i ($i = 1, ...b$) until convergence as a function of V_1: $V_i(s) = f(V_1(s_0), V_1(s_1), ...)$. If we add a point option

[3]Formally, d satisfies the triangle inequality, but does not satisfy the symmetry and the indiscernibles.



(a) Options discovered by A-MIMO

$s \setminus s'$	s_1	s_2	s_3	s_4	s_5	s_6
s_1	0	1	3	3	2	3
s_2	2	0	2	2	1	2
s_3	3	3	0	1	2	3
s_4	2	2	2	0	1	2
s_5	1	1	1	1	0	1
s_6	0	0	0	0	0	0

(b) $d_0(s, s')$

Figure 3: (a) Options discovered by A-MIMO with $k = 2$ are denoted by the dashed lines. (b) $d_0(s, s')$ for Figure 3a

from s' to g, then $V_1(s') = V^*(s')$. Thus, $d(s, s')$ is the smallest i where $V_i(s)$ reaches ϵ-optimal if we replace $V_1(s')$ with $V^*(s')$ when computing $V_i(s)$ as a function of V_1.

Example. We use the MDP shown in Figure 3a as an example. Consider the problem of finding a set of options so that the MDP can be solved within 2 iterations. We generate an instance of a set-cover optimization problem. The set of elements for the set cover is the set of states of the MDP that do not reach their optimal value within ℓ steps without any options $\mathcal{S} \setminus X_g^+$. Here, we denote a set of nodes that can be solved within ℓ steps by X_g^+. In the example, $\mathcal{U} = S \setminus X_g^+ = \{s_1, s_2, s_3, s_4\}$. A state s is included in a subset $X_{s'}$ iff $d(s, s') \leq \ell - 1$. For example, $X_{s_1} = \{s_1\}, X_{s_2} = \{s_1, s_2\}$. Thus, the set of subsets are given as: $X_{s_1} = \{s_1\}, X_{s_2} = \{s_1, s_2\}, X_{s_3} = \{s_3\}, X_{s_4} = \{s_3, s_4\}$. In this case, the approximation algorithm finds the optimal solution $\mathcal{C} = \{X_{s_2}, X_{s_4}\}$ for the set-cover optimization problem $(\mathcal{U}, \mathcal{X})$. We generate a point option for each state in $\mathcal{C}$. Thus, the output of the algorithm is a set of two point options from s_2 and s_4 to g. ◇

Theorem 5. A-MOMI *has the following properties:*

1. A-MOMI *runs in polynomial time.*
2. *It guarantees that the MDP is solved within ℓ iterations using the option set acquired by* A-MOMI $\mathcal{O}$.
3. *If the MDP is deterministic, the option set is at most $O(\log n)$ times larger than the smallest option set possible to solve the MDP within ℓ iterations.*

Proof. See the supplementary material. □

Note that the approximation bound for a deterministic MDP will inherent any improvements to the approximation algorithm for set cover. Set cover is known to be *NP*-hard to approximate up to a factor of $(1 - o(1)) \log n$ (Dinur & Steurer, 2014), thus there may be an improvement on the approximation ratio for the set cover problem, which will also improve the approximation ratio of A-MOMI.

4.2. A-MIMO

The outline of the approximation algorithm for MIMO (A-MIMO) is as follows.

1. Compute $d_\epsilon(s, s') : S \times S \rightarrow \mathbb{N}$ for each pair of states.
2. Using this distance function, solve an asymmetric k-center problem, which finds a set of center states that minimizes the maximum number of iterations for every state to converge.
3. Generate point options with initiation states set to the center states in the solution of the asymmetric k-center, and termination states set to the goal.

As in A-MOMI, we first compute the function d. Then, we exploit this characteristic of d and solve the asymmetric k-center problem (Panigrahy & Vishwanathan, 1998) on $(\mathcal{U}, d, k)$ to get a set of centers, which we use as initiation states for point options. The asymmetric k-center problem is a generalization of the metric k-center problem where the function d obeys the triangle inequality, but is not necessarily symmetric:

Definition 12 (AsymKCenter):
Given *a set of elements $\mathcal{U}$, a function $d : \mathcal{U} \times \mathcal{U} \rightarrow \mathbb{N}$, and an integer k,* **return** *$\mathcal{C}$ that minimizes $P(\mathcal{C}) = \max_{s \in U} \min_{c \in \mathcal{C}} d(s, c)$ subject to $|\mathcal{C}| \leq k$.*

We solve the problem using a polynomial-time approximation algorithm proposed by Archer (2001). The algorithm has a suboptimality bound of $O(\log^* k)$[4] where $k < |\mathcal{U}|$. It is known that the problem cannot be solved within a factor of $\log^* |\mathcal{U}| - \theta(1)$ unless $P = NP$ (Chuzhoy et al., 2005). As the procedure by Archer (2001) often finds a set of options smaller than k, we generate the rest of the options by greedily adding $\lfloor \log k \rfloor$ options at once. See the supplementary material for details. Finally, we generate a set of point options with initiation-states set to one of the centers and the termination state set to the goal state of the MDP. That is, for every c in $\mathcal{C}$, we generate a point option starting from c to the goal state g.

[4] $\log^*$ is the number of times the logarithm function must be iteratively applied before the result is less than or equal to 1.

Example. Consider an MDP shown in Figure 3a. The distance d_0 for the MDP is shown in Table 3b. Note that $d(s, s') \leq d(s, g)$ holds for every s, s' pair. Let us first consider finding one option ($k = 1$). This process corresponds to finding a column with the smallest maximum value in the Table 3b. The optimal point option is from s_5 to g as it has the smallest maximum value in the column. If $k = 2$, an optimal set of options is from s_2 and s_4 to g. Note that the optimal option for $k = 1$ is not in the optimal option set of size 2. This example shows that the strategy of greedily adding options does not find the optimal set. In fact, the improvement $L_{\epsilon,V_0}(\emptyset) - L_{\epsilon,V_0}(\mathcal{O})$ on by the greedy algorithm can be arbitrary small (i.e. 0) compared to the optimal option (see Proposition 1 in the supplementary material for a proof). $\diamond$

Theorem 6. A-MIMO *has the following properties:*

1. A-MIMO *runs in polynomial time.*
2. *If the MDP is deterministic, it has a bounded suboptimality of* $O(\log^* k)$.
3. *The number of iterations to solve the MDP using the option set acquired is upper bounded by* $P(\mathcal{C})$.

Proof. See the supplementary material. $\square$

5. Experiments

We evaluate the performance of the value-iteration algorithm using options generated by the approximation algorithms on several grid-based simple domains.

We ran the experiments on an 11×11 four-room domain and a 9×9 grid world with no walls. In both domains, the agent's goal is to reach a specific square. The agent can move in the usual four directions but cannot cross walls.

Visualizations: First, we visualize a variety of option types, including the optimal point options, those found by our approximation algorithms, and several option types proposed in the literature. We computed the optimal set of point options by enumerating every possible set of point options and picking the best. As an optimal set of options is not unique, we picked one arbitrarily. We are only able to find optimal solutions up to four options within 10 minutes, while the approximation algorithm could find any number of options within a few minutes. For eigenoptions, we ignored the eigenvector corresponding to the smallest eigenvalue ($\lambda_0 = 0$) in the graph Laplacian because it has a constant value for every state. Both betweenness options and eigenoptions are polynomial time algorithm, thus run in a few minutes. Figure 4 shows the optimal and bounded suboptimal set of options computed by A-MIMO. See the supplementary material for visualizations for the 9×9 grid domain.

Figure 4e shows the four bottleneck states with highest shortest-path betweenness centrality in the state-transition graph (Şimşek & Barto, 2009). Interestingly, the optimal options are quite close to the bottleneck states in the four-room domain, suggesting that bottleneck states are also useful for planning as a heuristic to find important subgoals.

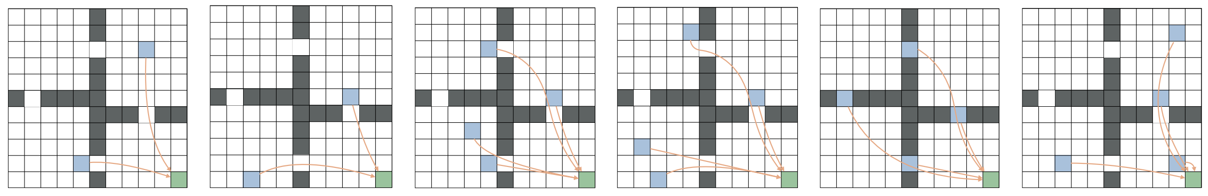
Figure 4f shows the set of subgoals discovered by graph Laplacian analysis following Machado et al. (2017). While they proposed to generate options to travel between subgoals for reinforcement learning, we generate a set of point options from each subgoal to the goal state as that is a better in the planning setting.

Quantitative Evaluation: Next, we run value iteration using the set of options generated by A-MIMO and A-MOMI. Figures 5a and 5b show the number of iterations on the four-room and the 9×9 grids using k options. The experimental results suggest that the suboptimal algorithm finds set of options similar to, but not quite as good as, the optimal ones. For betweenness options and eigenoptions, we evaluated every subset of options among the four and present results for the best subset found. Because betweenness options are placed close to the optimal options, the performance is close to optimal especially when the number of options is small.

In addition, we used A-MOMI to find a minimum option set to solve the MDP within the given number of iterations. Figures 5c and 5d show the number of options generated by A-MOMI compared to the minimum number of options.

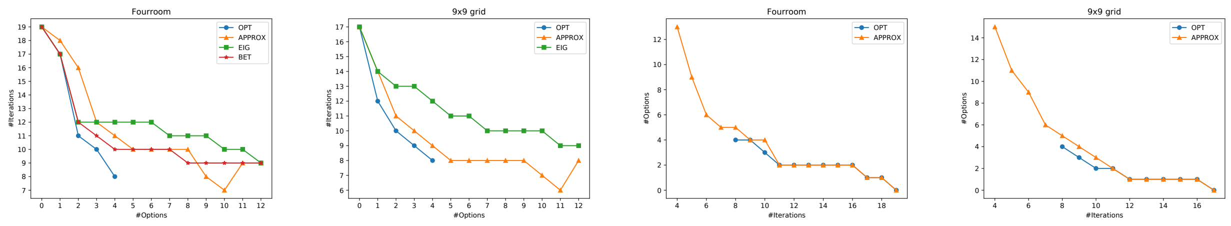
6. Related Work

Many heuristic algorithms have proposed to discover options (Iba, 1989; McGovern & Barto, 2001; Menache et al., 2002; Stolle & Precup, 2002; Şimşek & Barto, 2004; Şimşek & Barto, 2009; Konidaris & Barto, 2009; Machado et al., 2017; Eysenbach et al., 2019). For example, some investigate the use of bottleneck states (Stolle & Precup, 2002; Şimşek & Barto, 2009; Menache et al., 2002; Lehnert et al., 2018). Stolle & Precup (2002) proposed to set states with high visitation counts as subgoal states, which identifies bottleneck states in the four-room domain. Şimşek & Barto (2009) generalized the concept of a bottleneck to (shortest-path) betweenness of the graph to capture how pivotal the state is. Menache et al. (2002) used a learned model of the environment to run a Max-Flow/Min-Cut algorithm to the state-space graph to identify bottleneck states. These methods generate options that leverage the idea that subgoals are states visited most frequently. On the other hand, Şimşek & Barto (2004) proposed to generate options to encourage exploration by generating options to relatively novel states. Eysenbach et al. (2019) instead proposed learning a policy for each option so that the diversity of the trajectories by the set of options are maximized. These methods



(a) optimal $k = 2$ (b) approx. $k = 2$ (c) optimal $k = 4$ (d) approx. $k = 4$ (e) Betweenness (f) Eigenoptions

Figure 4: Comparison of the optimal point options with options generated by the approximation algorithm A-MIMO. The green square represents the termination state and the blue squares the initiation states. Observe that the approximation algorithm is similar to that of optimal options. Note that the optimal option set is not unique: there can be multiple optimal option sets, and we visualize the one returned by the solver.



(a) Four Room (MIMO) (b) 9×9 grid (MIMO) (c) Four Room (MOMI) (d) 9×9 grid (MOMI)

Figure 5: MIMO and MOMI evaluations. Parts (a)–(b) show the number of iterations for VI using options generated by A-MIMO. Parts (c)–(d) show the number of options generated by A-MOMI to ensure the MDP is solved within a given number of iterations. OPT: optimal set of options. APPROX: a bounded suboptimal set of options generated by A-MIMO an A-MOMI. BET: betweenness options. EIG: eigenoptions.

generate options to explore infrequently visited states.

While empirical results show that these algorithms are useful in some scenarios, the conditions under which the methods are effective is often unclear because the relationship between the objective of the skill discovery algorithm and that of the agent is often not established. In fact, Jong et al. (2008) sought to investigate the utility of skills empirically and pointed out that introducing skills might worsen the learning performance. Harb et al. (2017) proposed to formulate good options to be options which minimize the deliberation costs in the bounded rationality framework (Simon, 1957). Brunskill & Li (2014) targeted the lifelong reinforcement learning setting and proposed an option generation method for lifelong reinforcement learning. They analyzed the sample complexity of RMAX using options and proposed an option discovery targeting to minimize the sample complexity. Solway et al. (2014) formalized an optimal behavioral hierarchy as a model which fits the behavior of the agent in tasks the best.

For planning, several works have shown empirically that adding a particular set of options or macro-operators can speed up planning algorithms (Francis & Ram, 1993; Sutton & Barto, 1998; Silver & Ciosek, 2012; Konidaris, 2016). Mann et al. (2015) analyzed the convergence rate of approximate value iteration with and without options, and showed that options lead to faster convergence if their durations are longer and the value function is initialized pessimistically.

7. Conclusions

We considered two fundamental theoretical questions concerning the use of behavioral abstractions to solve MDPs: (1) minimize the size of option set given a maximum number of iterations (MOMI) and (2) minimize the number of iterations given a maximum size of option set (MIMO). We showed that both problems are computationally intractable, even for deterministic MDPs. For each problem, we produced a polynomial-time algorithm for MDPs with bounded reward and goal states, and with bounded optimality for deterministic MDPs. Although these algorithms are not practical for a single-task planning, we believe they may be a useful foundation for future option discovery methods. In the future, we are interested in using the insights established here to develop principled option-discovery algorithms for model-based reinforcement learning. Since we now know which options minimize planning time, we can better guide model-based agents toward learning them and potentially reduce sample complexity considerably.

Acknowledgments

We would like to thank the anonymous reviewer for their advice and suggestions to improve the inapproximability result for MOMI. We would like to thank Maehara Takanori and Kazuki Yoshizoe for their advice on formulating the problems. This work is supported in part by an AFOSR Young Investigator Grant to George Konidaris, under agreement number FA9550-17-1-0124. D Ellis Hershkowitz is supported in part by NSF grants CCF-1618280, CCF-1814603, CCF-1527110 and NSF CAREER award CCF-1750808.

References

Archer, A. Two O(log* k)-approximation algorithms for the asymmetric k-center problem. In *International Conference on Integer Programming and Combinatorial Optimization*, pp. 1–14, 2001.

Bacon, P.-L. On the bottleneck concept for options discovery. *Masters thesis, McGill University*, 2013.

Bakker, B. and Schmidhuber, J. Hierarchical reinforcement learning based on subgoal discovery and subpolicy specialization. In *Proceedings of the 8th Conference on Intelligent Autonomous Systems*, pp. 438–445, 2004.

Brunskill, E. and Li, L. PAC-inspired option discovery in lifelong reinforcement learning. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*, pp. 316–324, 2014.

Chuzhoy, J., Guha, S., Halperin, E., Khanna, S., Kortsarz, G., Krauthgamer, R., and Naor, J. S. Asymmetric k-center is log* n-hard to approximate. *Journal of the ACM*, 52 (4):538–551, 2005.

Chvatal, V. A greedy heuristic for the set-covering problem. *Mathematics of operations research*, 4(3):233–235, 1979.

Dinitz, M., Kortsarz, G., and Raz, R. Label cover instances with large girth and the hardness of approximating basic k-spanner. In *International Colloquium on Automata, Languages, and Programming*, pp. 290–301. Springer, 2012.

Dinur, I. and Steurer, D. Analytical approach to parallel repetition. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pp. 624–633. ACM, 2014.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. In *Proceedings of the Seventh International Conference on Learning Representations*, 2019.

Francis, A. G. and Ram, A. The utility problem in case-based reasoning. In *Case-Based Reasoning: Papers from the 1993 Workshop*, pp. 160–161, 1993.

Goldsmith, J., Littman, M. L., and Mundhenk, M. The complexity of plan existence and evaluation in probabilistic domains. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–97)*, pp. 182–189, San Francisco, CA, 1997. Morgan Kaufmann Publishers.

Harb, J., Bacon, P.-L., Klissarov, M., and Precup, D. When waiting is not an option: Learning options with a deliberation cost. *arXiv preprint arXiv:1709.04571*, 2017.

Iba, G. A. A heuristic approach to the discovery of macro-operators. *Machine Learning*, 3(4):285–317, 1989.

Jong, N. K., Hester, T., and Stone, P. The utility of temporal abstraction in reinforcement learning. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 299–306, 2008.

Konidaris, G. Constructing abstraction hierarchies using a skill-symbol loop. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, volume 2016, pp. 1648, 2016.

Konidaris, G. and Barto, A. Skill discovery in continuous reinforcement learning domains using skill chaining. In *Advances in Neural Information Processing Systems*, pp. 1015–1023, 2009.

Lehnert, L., Laroche, R., and van Seijen, H. On value function representation of long horizon problems. In *AAAI*, 2018.

Littman, M. L. Probabilistic propositional planning: Representations and complexity. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pp. 748–754. AAAI Press/The MIT Press, 1997.

Littman, M. L., Dean, T. L., and Kaelbling, L. P. On the complexity of solving Markov decision problems. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pp. 394–402, 1995.

Machado, M. C., Bellemare, M. G., and Bowling, M. A Laplacian framework for option discovery in reinforcement learning. In *Proceedings of the Thirty-fourth International Conference on Machine Learning*, 2017.

Mann, T. and Mannor, S. Scaling up approximate value iteration with options: Better policies with fewer iterations. In *International Conference on Machine Learning*, pp. 127–135, 2014.

Mann, T. A., Mannor, S., and Precup, D. Approximate value iteration with temporally extended actions. *Journal of Artificial Intelligence Research*, 53:375–438, 2015.

McGovern, A. and Barto, A. G. Automatic discovery of subgoals in reinforcement learning using diverse density. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 361–368, 2001.

Menache, I., Mannor, S., and Shimkin, N. Q-cut - dynamic discovery of sub-goals in reinforcement learning. In *European Conference on Machine Learning*, pp. 295–306, 2002.

Moradi, P., Shiri, M. E., Rad, A. A., Khadivi, A., and Hasler, M. Automatic skill acquisition in reinforcement learning using graph centrality measures. *Intelligent Data Analysis*, 16(1):113–135, 2012.

Panigrahy, R. and Vishwanathan, S. An O(log* n) approximation algorithm for the asymmetric p-center problem. *Journal of Algorithms*, 27(2):259–268, 1998.

Papadimitriou, C. H. and Tsitsiklis, J. N. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, August 1987.

Pickett, M. and Barto, A. Policyblocks: An algorithm for creating useful macro-actions in reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pp. 506–513, 2002.

Precup, D. and Sutton, R. S. Multi-time models for temporally abstract planning. In *Advances in neural information processing systems*, pp. 1050–1056, 1998.

Puterman, M. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994.

Silver, D. and Ciosek, K. Compositional planning using optimal option models. In *Proceedings of the 29th International Conference on Machine Learning*, pp. 1063–1070, 2012.

Simon, H. A. *Models of man; social and rational.* Wiley, 1957.

Şimşek, Ö. and Barto, A. Using relative novelty to identify useful temporal abstractions in reinforcement learning. In *Proceedings of the 21st International Conference on Machine Learning*, pp. 751–758, 2004.

Şimşek, Ö. and Barto, A. G. Skill characterization based on betweenness. In *Advances in Neural Information Processing Systems*, pp. 1497–1504, 2009.

Şimşek, Ö., Wolfe, A., and Barto, A. Identifying useful subgoals in reinforcement learning by local graph partitioning. In *Proceedings of the Twenty Second International Conference on Machine Learning*, pp. 816–823, 2005.

Solway, A., Diuk, C., Córdova, N., Yee, D., Barto, A. G., Niv, Y., and Botvinick, M. M. Optimal behavioral hierarchy. *PLoS computational biology*, 10(8):e1003779, 2014.

Stolle, M. and Precup, D. Learning options in reinforcement learning. In *International Symposium on Abstraction, Reformulation, and Approximation*, pp. 212–223, 2002.

Sutton, R., Precup, D., and Singh, S. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1): 181–211, 1999.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Williams, R. J. and Baird, L. C. Tight performance bounds on greedy policies based on imperfect value functions. Technical report, College of Computer Science, Northeastern University, 1993.