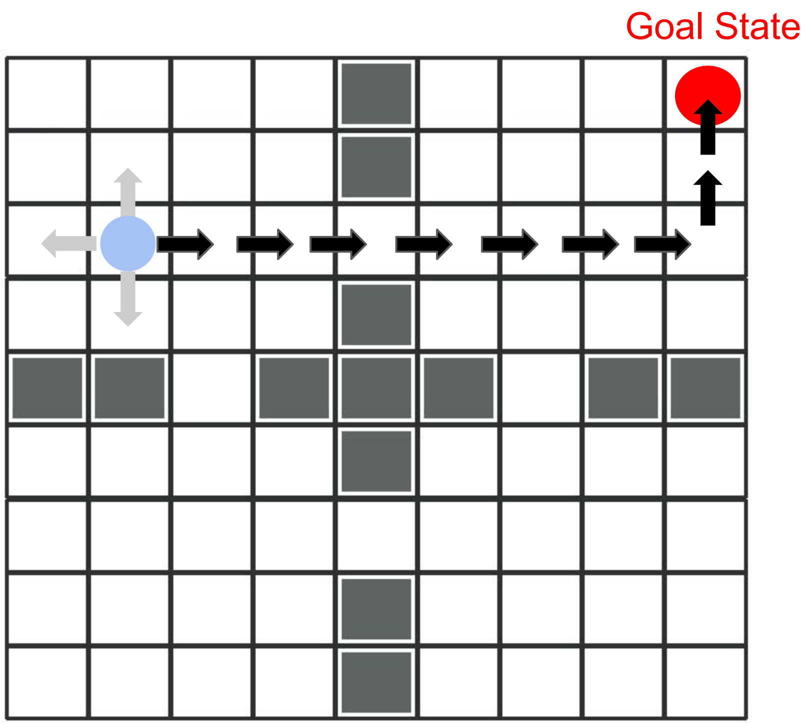
Finding Options that Minimize Planning Time

Yuu Jinnai*, David Abel, Michael L. Littman, George Konidaris

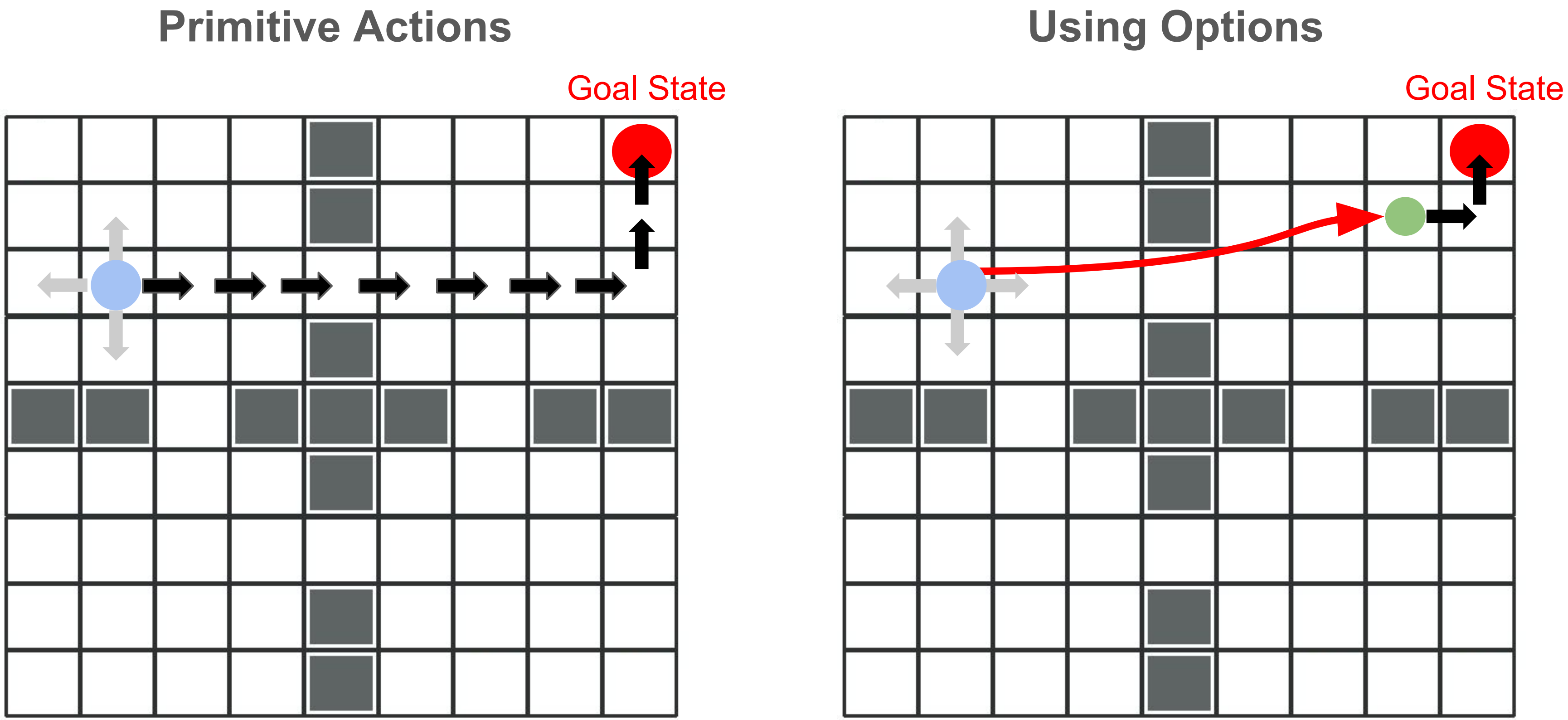
*: Presenter

Behavioral Hierarchy (Options) (Sutton et al. 1999)

Primitive Actions



Behavioral Hierarchy (Options) (Sutton et al. 1999)

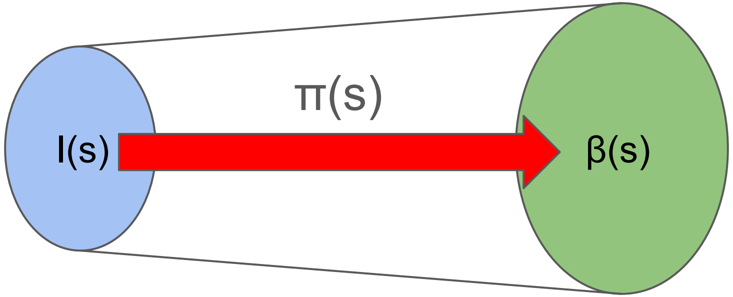


Options (Sutton et al. 1999)

I(s) := initiation set

β(s) := termination probability

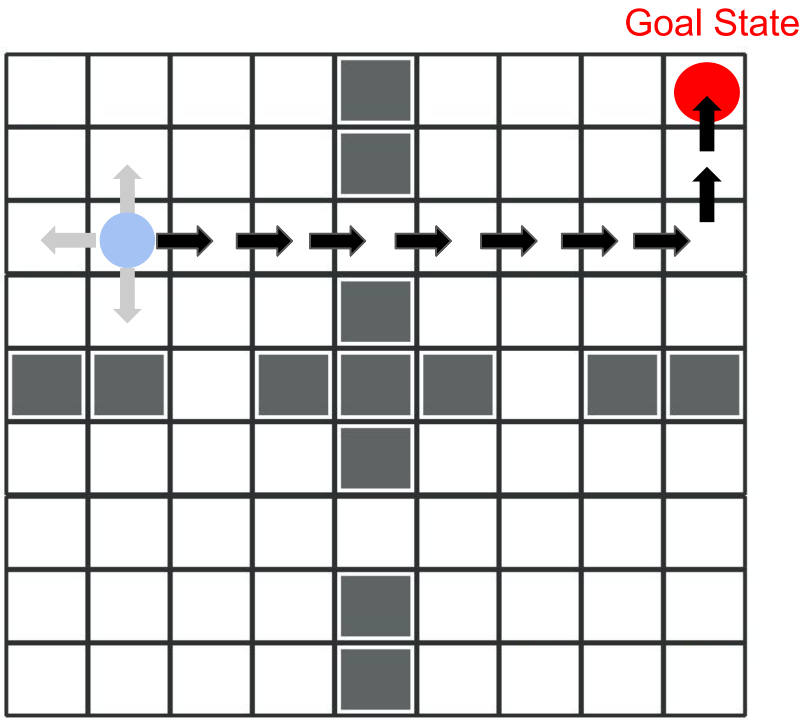
π(s) := policy



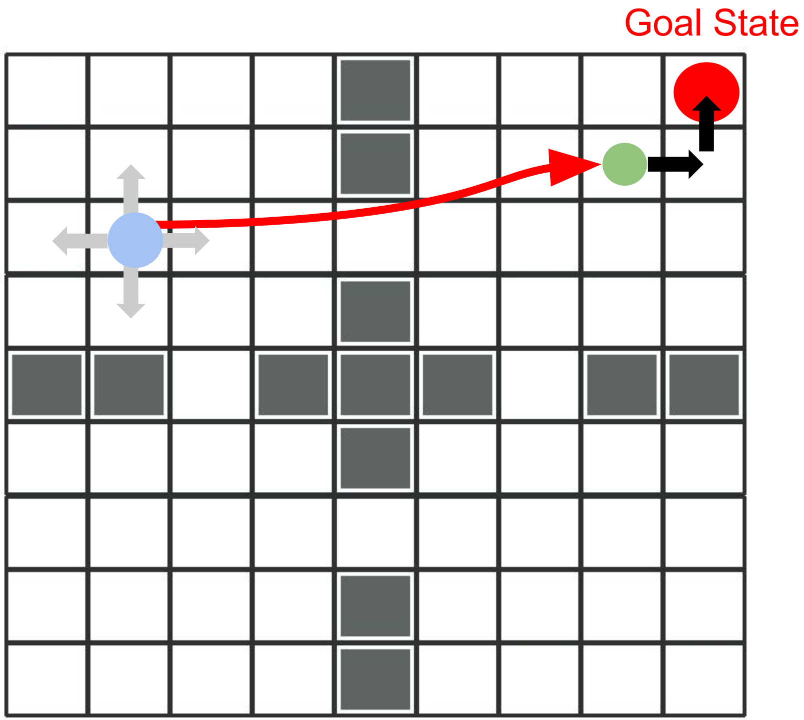
We assume the optimal policy is given

Behavioral Hierarchy (Options) (Sutton et al. 1999)

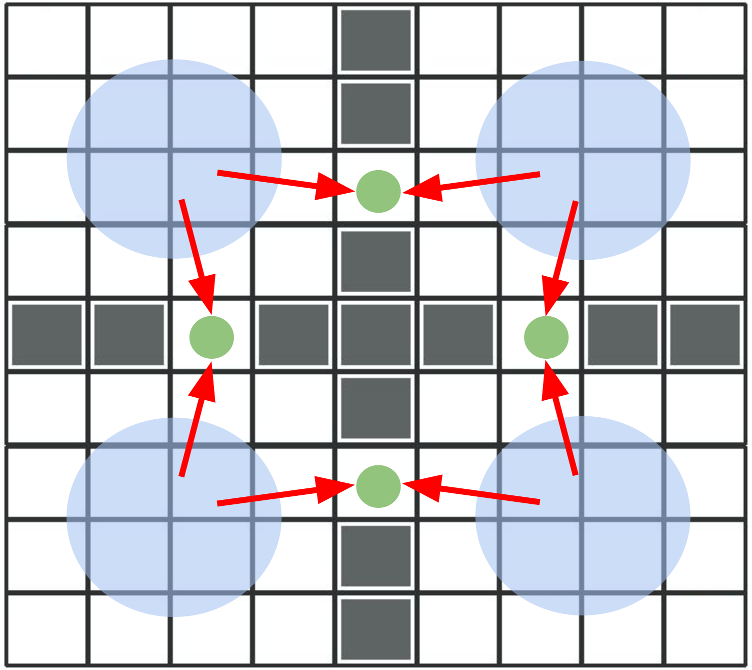
Primitive Actions



Using Options



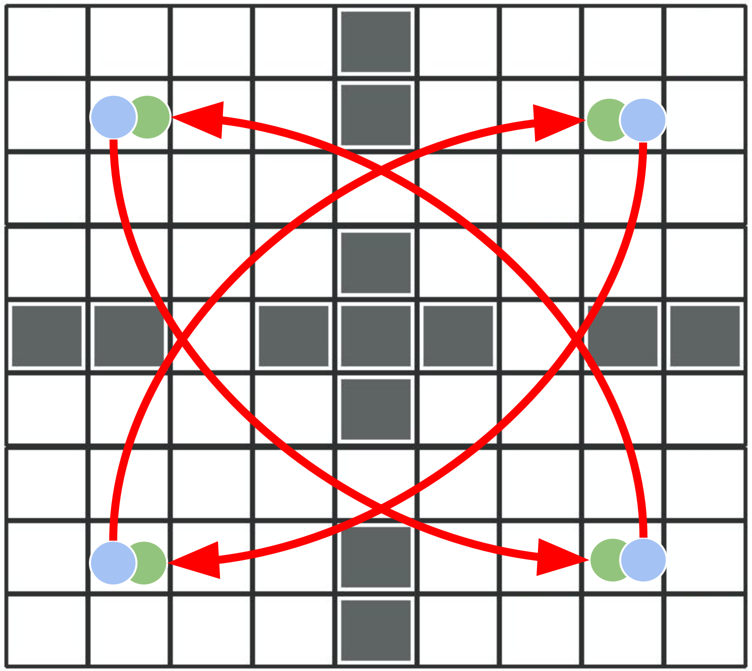
Bottleneck Options (Stolle&Precup 2002)



: Initiation State: *I*(s)

: Termination State: β(s)

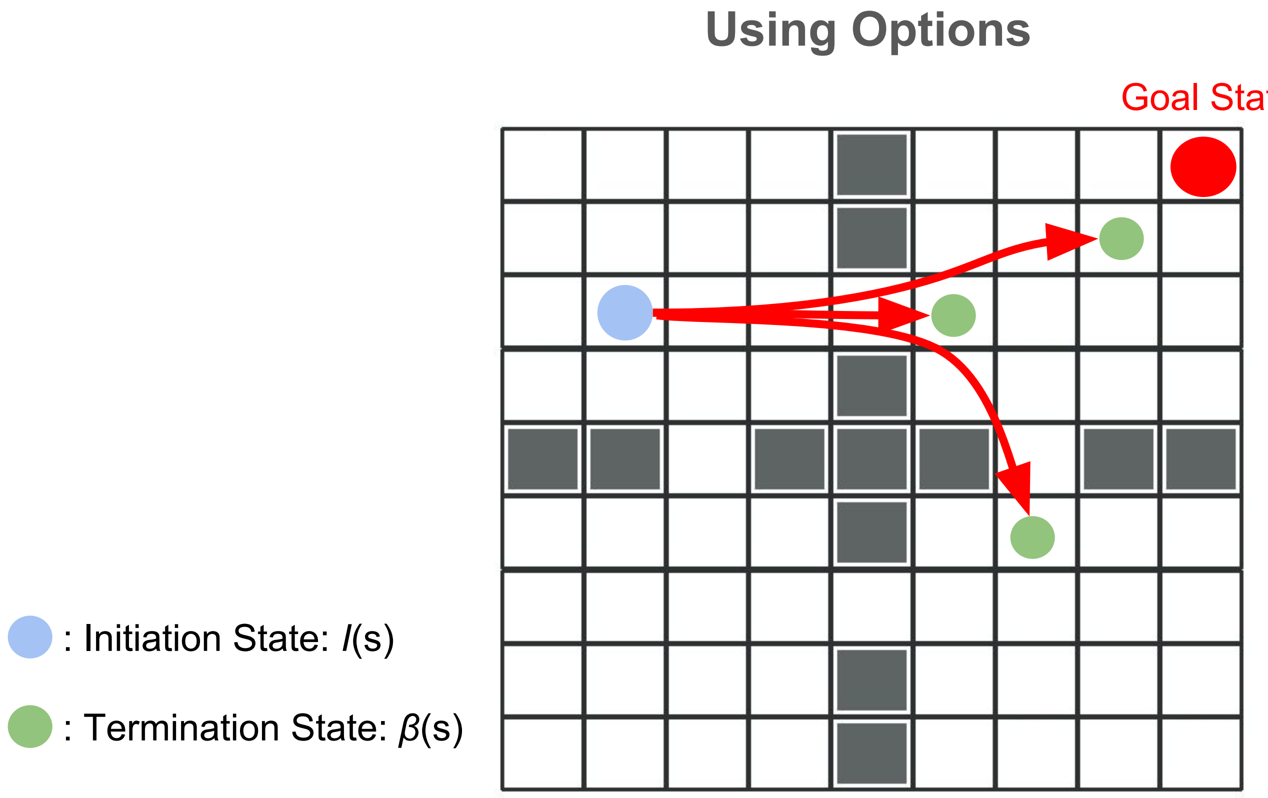
Graph-Theoretic Eigenoptions (Machado et al. 2017)



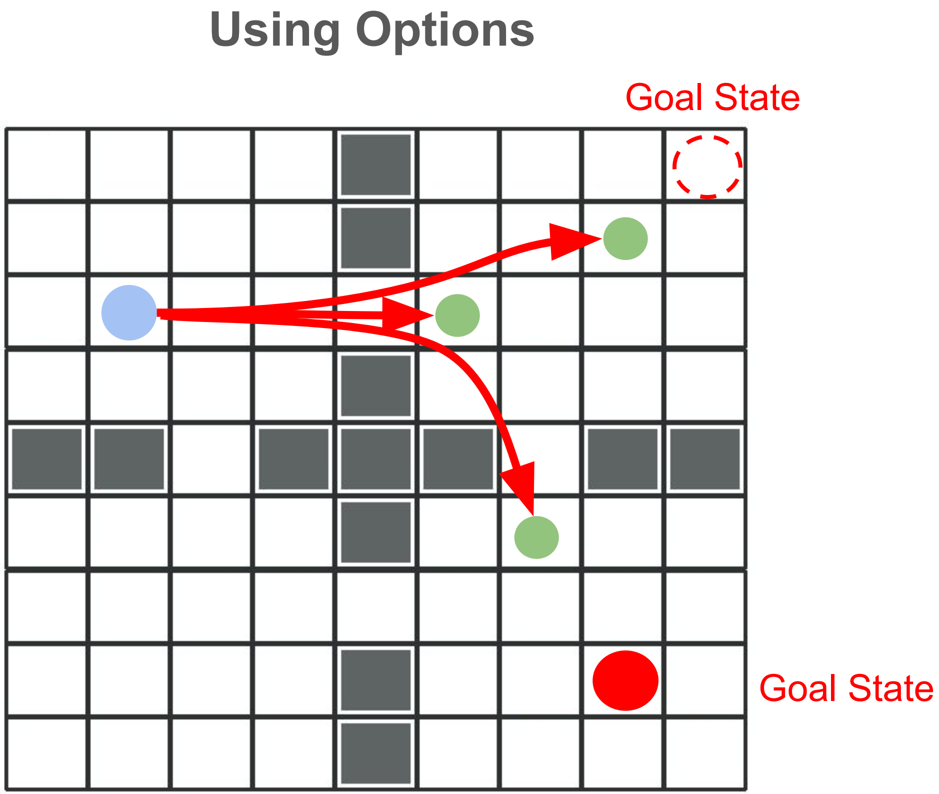
: Initiation State: I(s)

: Termination State: β(s)

Research Question: Which Options are the Best?



Research Question: Which Options are the Best?

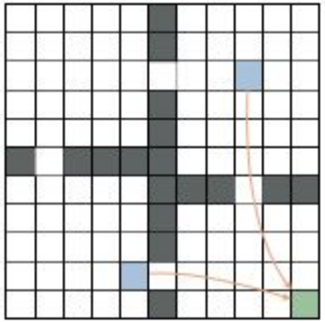


: Initiation State: $I(s)$

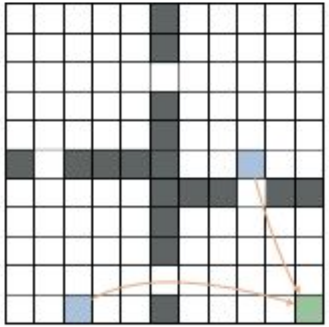
: Termination State: $\beta(s)$

Overview

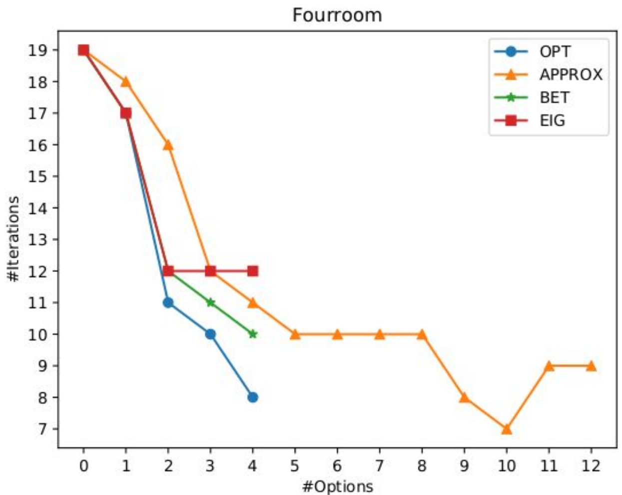
1. Definition of ‘optimal’ options for planning
2. The complexity of computing optimal options is NP-hard
3. Approximation algorithm for computing optimal options
4. Empirical evaluation to compare existing heuristic algorithms



(a) optimal $k = 2$



(b) approx. $k = 2$



: Initiation State: $I(s)$

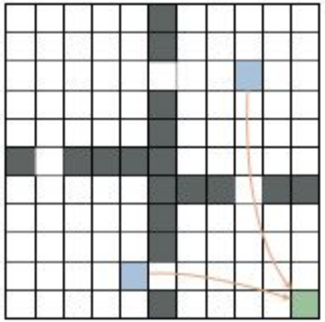
: Termination State: $\beta(s)$

Overview

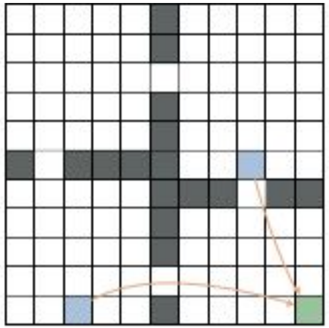
1. **Definition of ‘optimal’ options for planning**
2. **The complexity of computing optimal options is NP-hard**
3. Approximation algorithm for computing optimal options
4. Empirical evaluation to compare existing heuristic algorithms

: Initiation State: *I*(s)

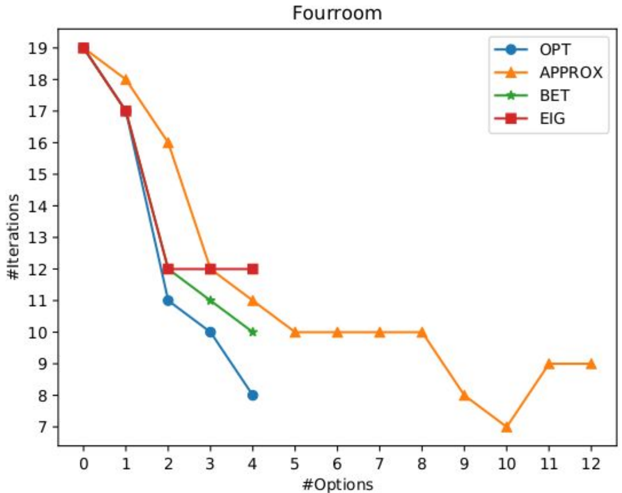
: Termination State: *β*(s)



(a) optimal $k = 2$



(b) approx. $k = 2$



Markov Decision Process (MDP)

$M = (S, A, T, R, \gamma)$

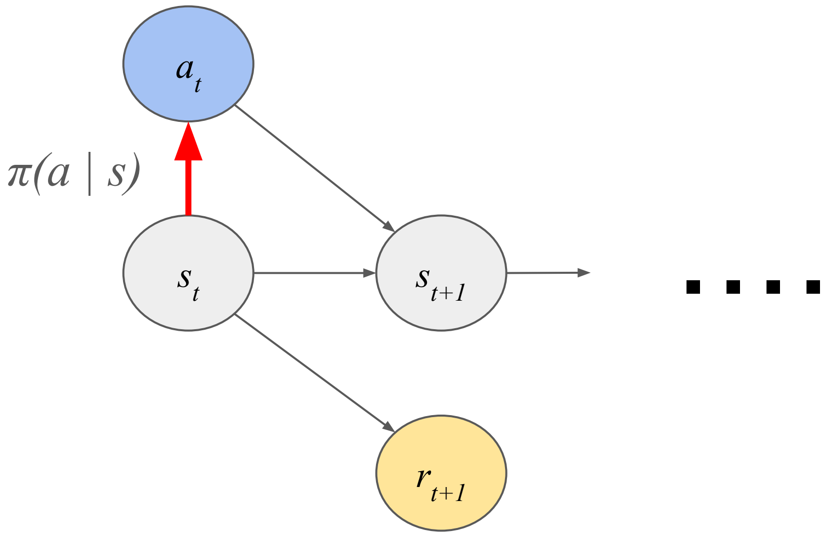
S: set of states

A: set of actions

T: transition function

R: reward function

γ: discount factor



Planning Problem

Compute the optimal value for **every** state

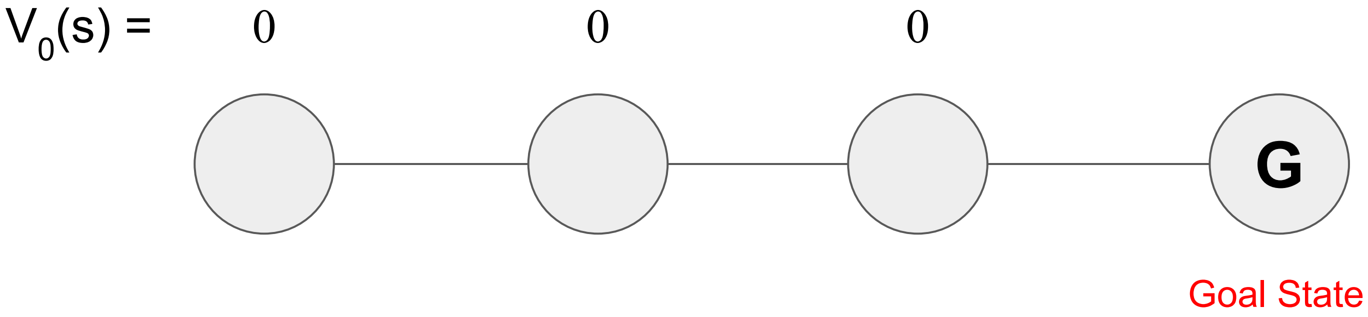
Definition 1 (Value-Planning Problem): **Given** *an MDP* $M = \langle \mathcal{S}, \mathcal{A}, R, T, \gamma \rangle$ *and a non-negative real-value* ϵ, **return** *a value function,* V *such that* $|V(s) - V^*(s)| < \epsilon$ *for all* $s \in \mathcal{S}$.

$V^*(s) =$ γ^3 γ^2 γ^1

Goal State

Value Iteration

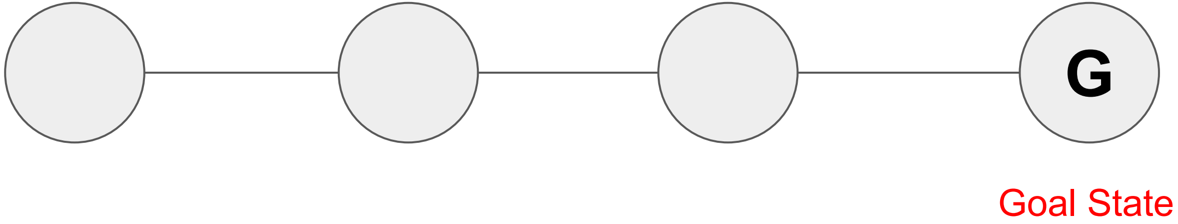
We focus on value iteration algorithm



Value Iteration

We focus on value iteration algorithm

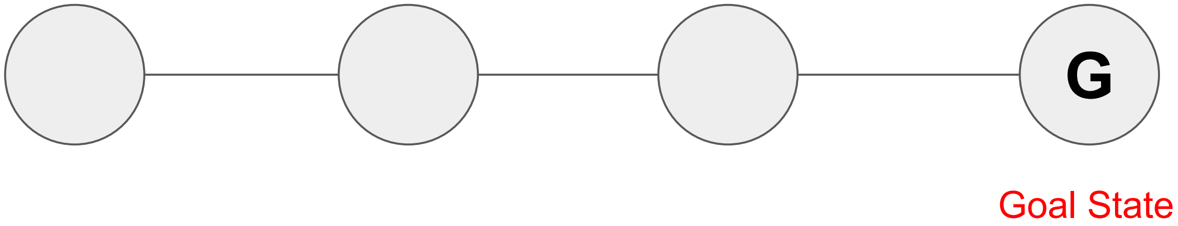
$V_1(s) =$	0	0	γ^1
$V_0(s) =$	0	0	0



Value Iteration

We focus on value iteration algorithm

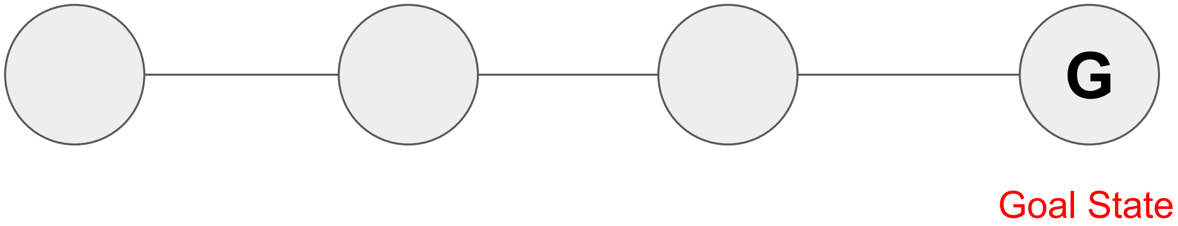
$V_2(s) =$	0	γ^2	γ^1
$V_1(s) =$	0	0	γ^1
$V_0(s) =$	0	0	0



Value Iteration

We focus on value iteration algorithm

$V_3(s) =$	γ^3	γ^2	γ^1
$V_2(s) =$	0	γ^2	γ^1
$V_1(s) =$	0	0	γ^1
$V_0(s) =$	0	0	0

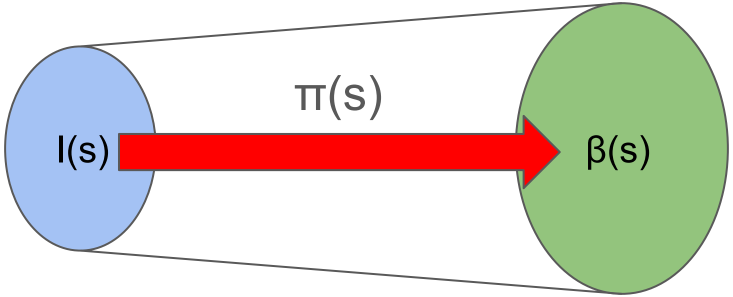


Options (Sutton et al. 1999)

I(s) := initiation set

β(s) := termination probability

π(s) := policy



We assume the optimal policy is given

Value Iteration with Options

Our objective is to speedup the value iteration algorithm by adding options to the primitive actions

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

Value Iteration with Options

Our objective is to speedup the value iteration algorithm by adding options to the primitive actions

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

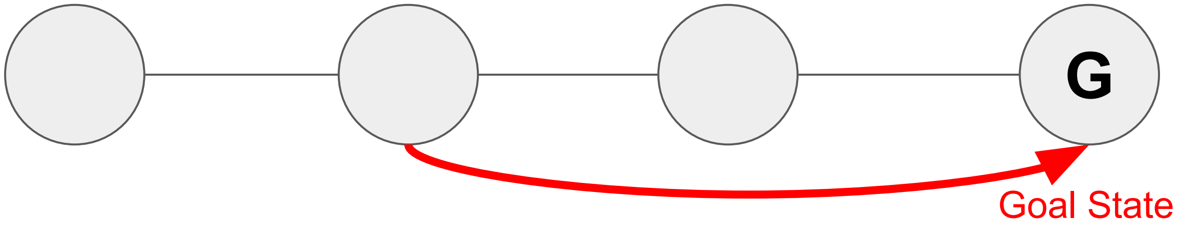
We assume that the model of the options are given to the agent

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

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

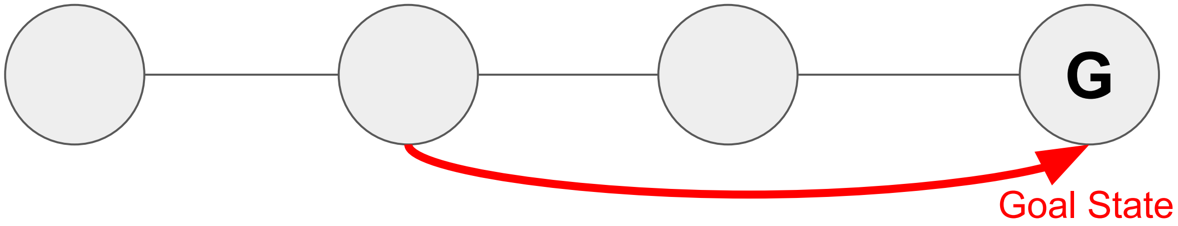
Value Iteration with Options

$V_0(s) =$ 0 0 0



Value Iteration with Options

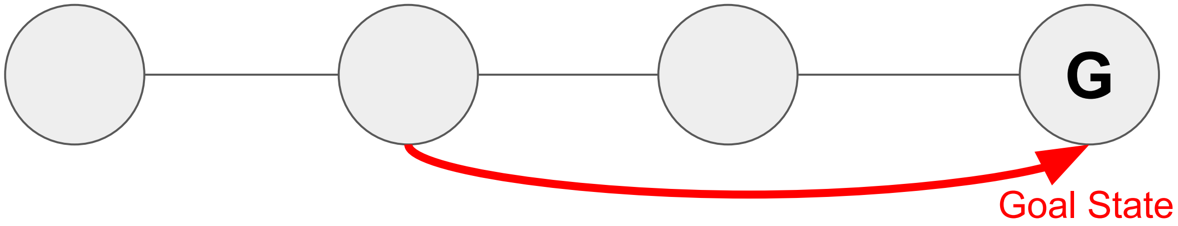
$V_1(s) =$	0	γ^2	γ^1
$V_0(s) =$	0	0	0



Value Iteration with Options

→It needs fewer iterations than just primitive actions!

$V_2(s) =$	γ^3	γ^2	γ^1
$V_1(s) =$	0	γ^2	γ^1
$V_0(s) =$	0	0	0



Objective Functions

1. **Number of Iterations:** $L(O)$
2. **Number of options:** $|O|$

Definition 3 ($L(\mathcal{O})$): *The number of iterations* $L(\mathcal{O})$ *of a value-iteration algorithm using option 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* $s \in \mathcal{S}$.

Two Optimization Problems: MIMO and MOMI

1. **Number of Iterations:** $L(O)$
2. **Number of options:** $|O|$

MIMO: Minimize $L(O)$ subject to $|O| \leqq k$

MOMI: Minimize $|O|$ subject to $L(O) \leqq L$

Two Optimization Problems: MIMO and MOMI

1. **Number of Iterations:** $L(O)$
2. **Number of options:** $|O|$

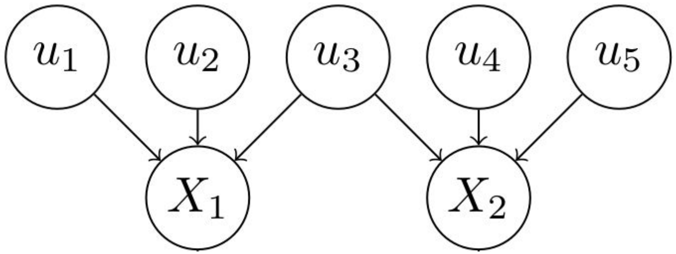
MIMO: Minimize $L(O)$ subject to $|O| \leqq k$

MOMI: Minimize $|O|$ subject to $L(O) \leqq L$

→**Both MIMO and MOMI are NP-hard.**

Proof: Reduction from the set cover decision problem.

NP-Hardness Proof Sketch



Reduction from the set cover decision problem.

Set Cover Decision Problem:

A set of elements $\{u_1, u_2, ...\} = \mathcal{U}$

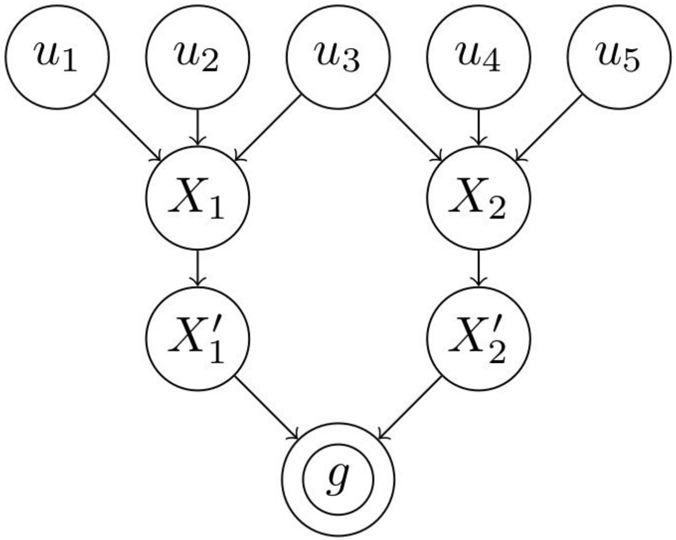
A set of subsets $\mathcal{X} = \{X_1, X_2, ...\}, X_i \subseteq \mathcal{U}$

Find a set of subsets $\mathcal{C} \subseteq \mathcal{X}$ of size $\boldsymbol{k}$, such that $\bigcup_{X \in \mathcal{C}} X = \mathcal{U}$

NP-Hardness Proof Sketch

Reduction:

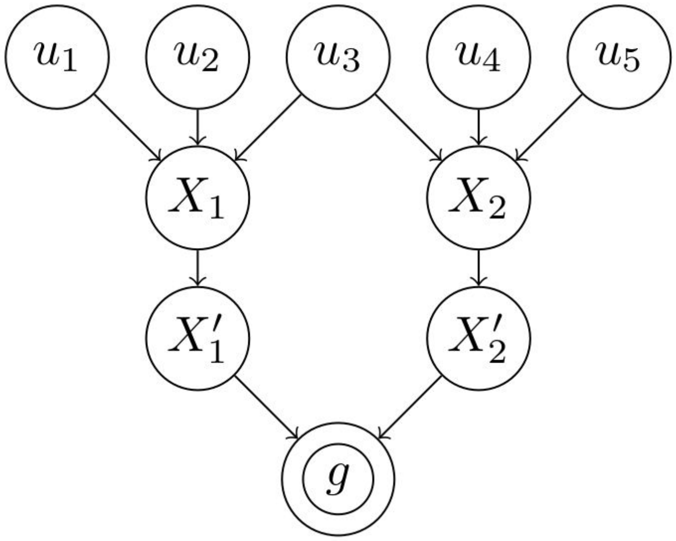
1. Construct a shortest path problem



NP-Hardness Proof Sketch

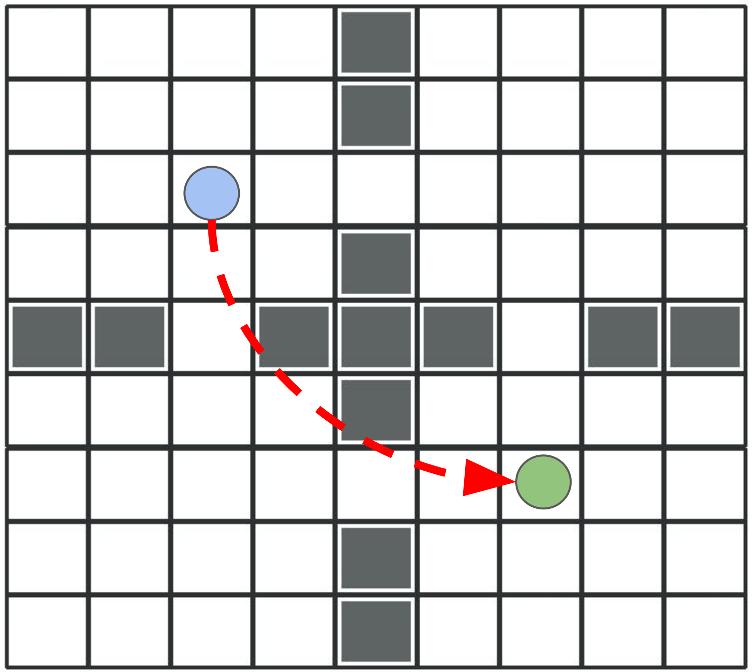
Reduction:

1. Construct a shortest path problem
2. Restrict to point options



Point Option

From one state to one state



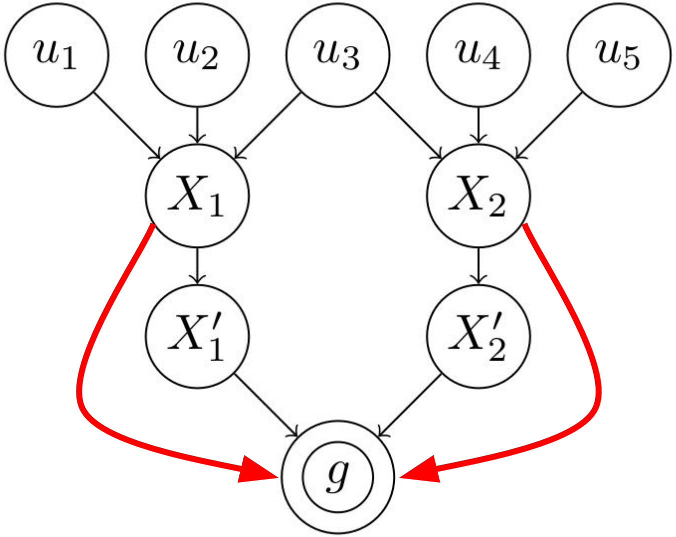
: Initiation State: $I(s)$

: Termination State: $\beta(s)$

NP-Hardness Proof Sketch

Reduction:

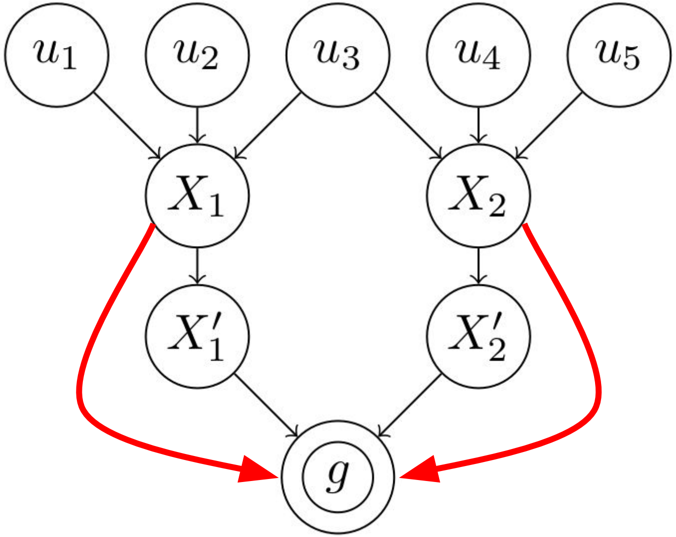
1. Construct a shortest path problem
2. Restrict to point options
3. Reduce to a decision problem of finding a set of point options such that $|O| \leqq k$ and $L(O) \leqq 2$



NP-Hardness Proof Sketch

Reduction:

1. Construct a shortest path problem
2. Restrict to point options
3. Reduce to a decision problem of finding a set of point options such that $|O| \leqq k$ and $L(O) \leqq 2$
4. MIMO / MOMI are optimization versions of the above problem, thus NP-hard.



Finding Optimal Behavioral Abstraction is NP-Hard

Given a single deterministic shortest-path problem, finding an optimal set of point options is NP-hard

Finding Optimal Behavioral Abstraction is NP-Hard

Given a set of MDPs, finding a set of options which minimizes an expected (or maximum) planning time is NP-hard.

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

Finding Optimal Behavioral Abstraction is NP-Hard

Given a set of MDPs, finding a set of options which minimizes an expected (or maximum) planning time is NP-hard.

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

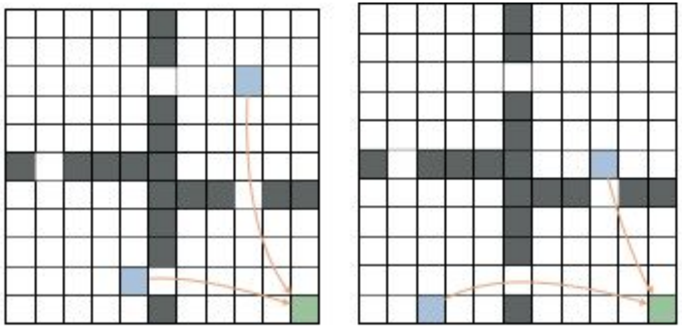
→Can we solve it approximately (with some guarantee) in polynomial time?

Overview

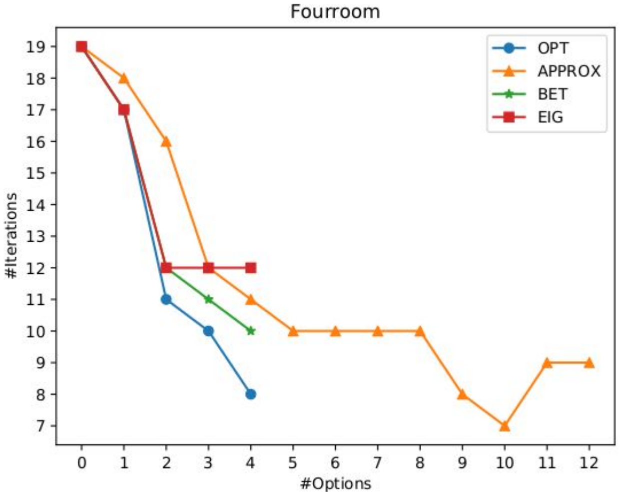
1. Definition of ‘optimal’ options for planning
2. The complexity of computing optimal options is NP-hard
3. **Approximation algorithm for computing optimal options**
4. Empirical evaluation to compare existing heuristic algorithms

: Initiation State: *I*(s)

: Termination State: *β*(s)



(a) optimal $k = 2$ (b) approx. $k = 2$



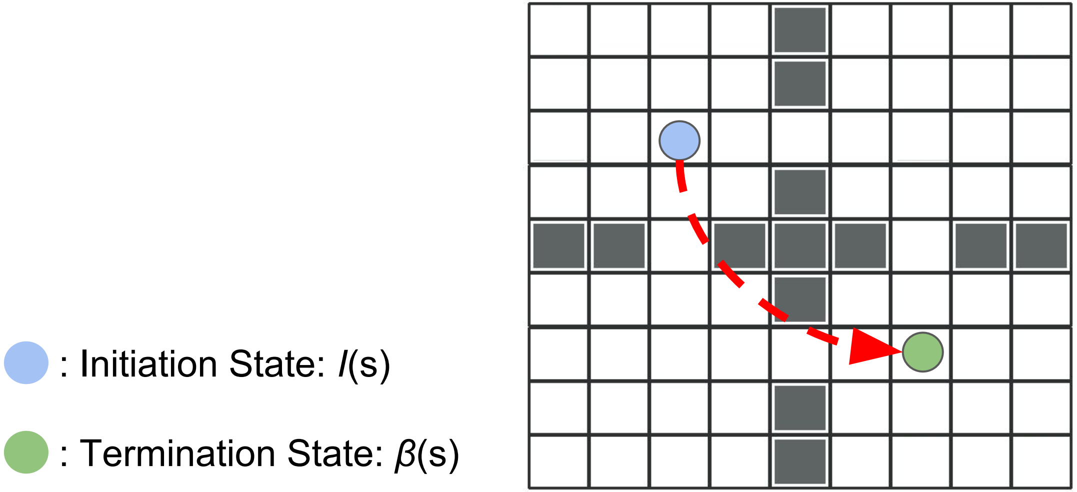
Assumptions

Task:

- Goal-based MDP: reach a goal while collecting maximum reward
- Maximum total reward is bounded

Options:

- Point options



Assumptions

The algorithms are not faster than solving the MDP itself

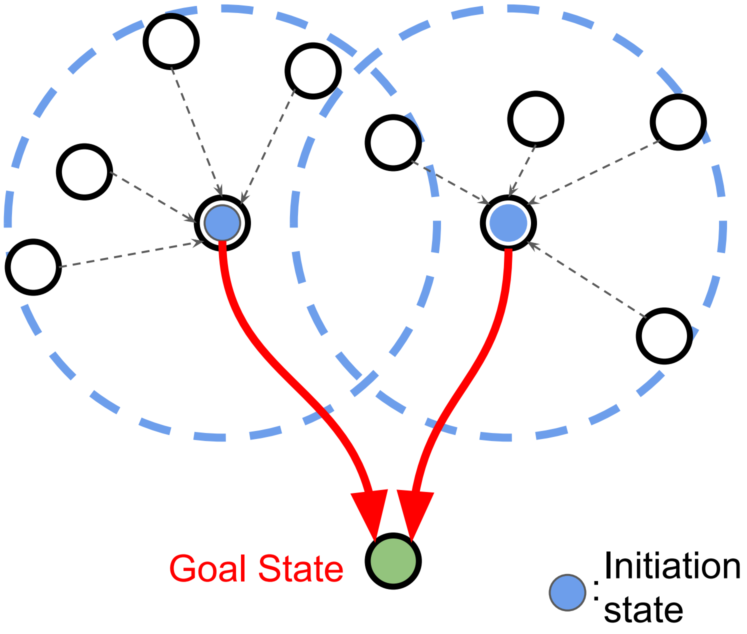
→The purpose of the algorithm is to analyze the quality of the heuristic algorithms, not for speeding up a single planning online

Approximation Algorithm for MIMO

MIMO: Minimize $L(O)$ subject to $|O| \leqq k$

Overview of the Algorithm:

1. Calculate the 'distance' between states
2. Reduce the asymmetric k-center problem based on the distance



Approximation Algorithm for MIMO

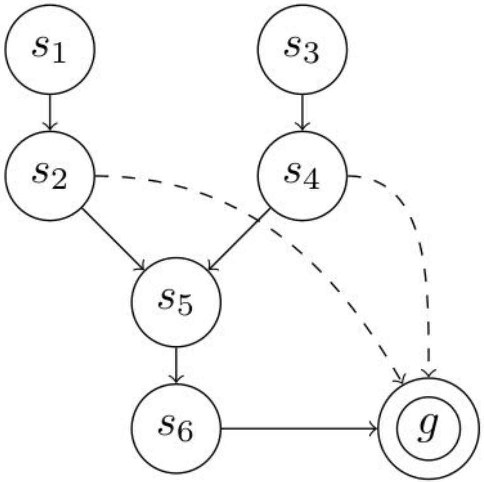
Theorem: The approximation algorithm has the following properties:

1. The algorithm runs in polynomial time.
2. The algorithm gives the upper bound of the number of iterations required to solve the MDP using the option set
3. If the MDP is deterministic, the number of iterations is at most a factor of O(log* k) times the minimum number of iterations, where k is the number of options allowed

Approximation Algorithm for MOMI

MOMI: Minimize $|O|$ subject to $L(O) \leqq L$

X_s: A set of states which can be solved within L steps if we add a point option from s to g



Example: L = 2

$$X_{s_1} = \{s_1\}$$
$$X_{s_2} = \{s_1, s_2\}$$
$$X_{s_3} = \{s_3\}$$
$$X_{s_4} = \{s_3, s_4\}$$

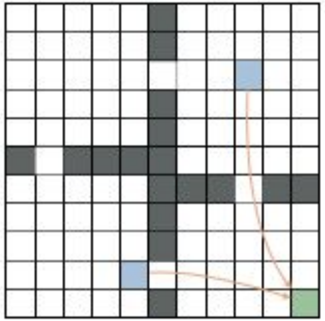
Approximation Algorithm for MOMI

Theorem: The approximation algorithm has the following properties:

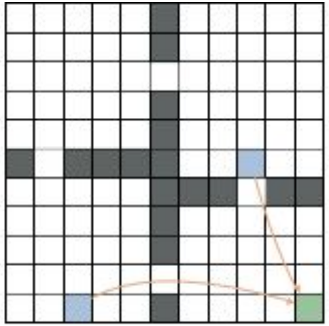
1. The algorithm runs in polynomial time.
2. The algorithm guarantees that the MDP is solved within the specified number of iterations using the acquired option set
3. If the MDP is deterministic, the option set is at most a factor of $O(\log |S|)$ times larger than the smallest option set

Overview

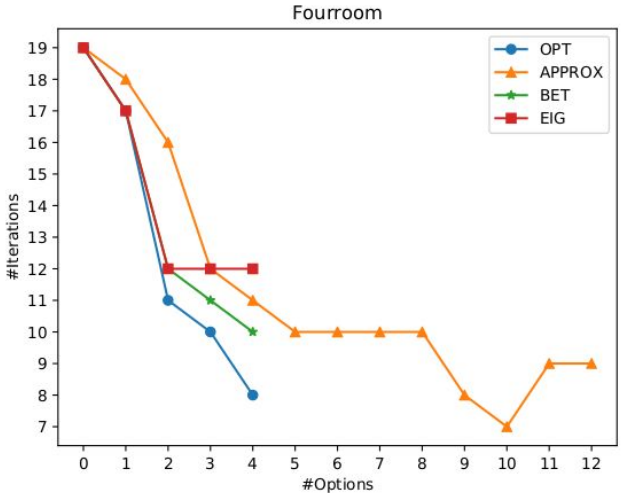
1. Definition of ‘optimal’ options for planning
2. The complexity of computing optimal options is NP-hard
3. Approximation algorithm for computing optimal options
4. **Empirical evaluation to compare existing heuristic algorithms**



(a) optimal $k = 2$



(b) approx. $k = 2$

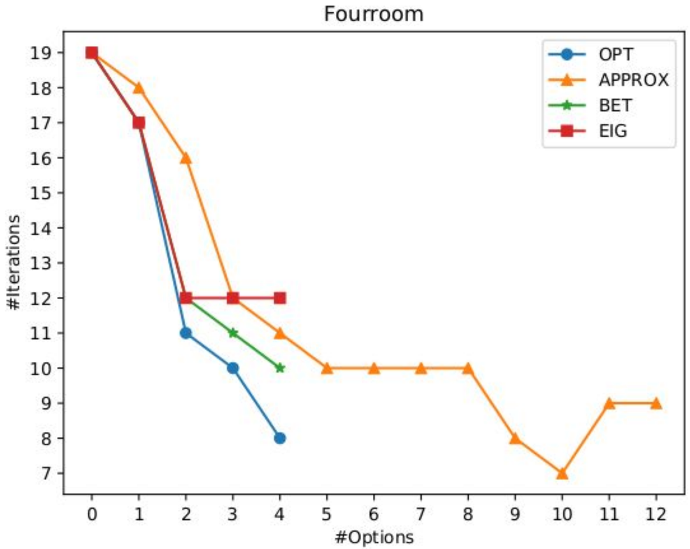


: Initiation State: $I(s)$

: Termination State: $\beta(s)$

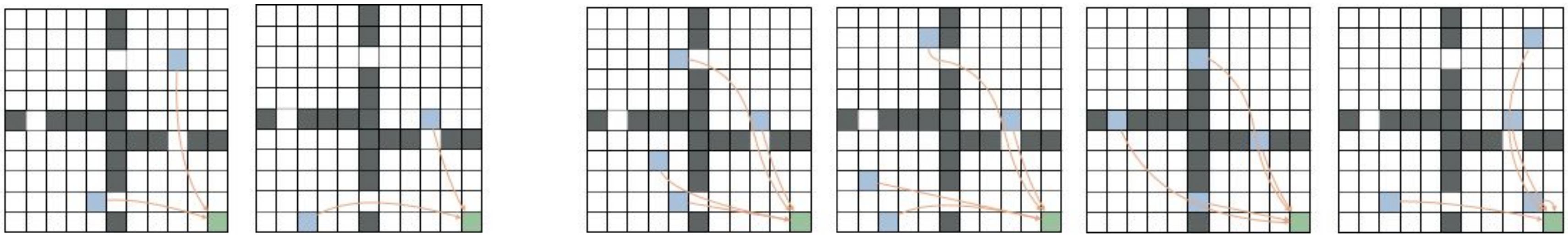
Evaluations: MIMO

MIMO: Minimize $L(O)$ subject to $|O| \leq k$



: Initiation State: I(s)

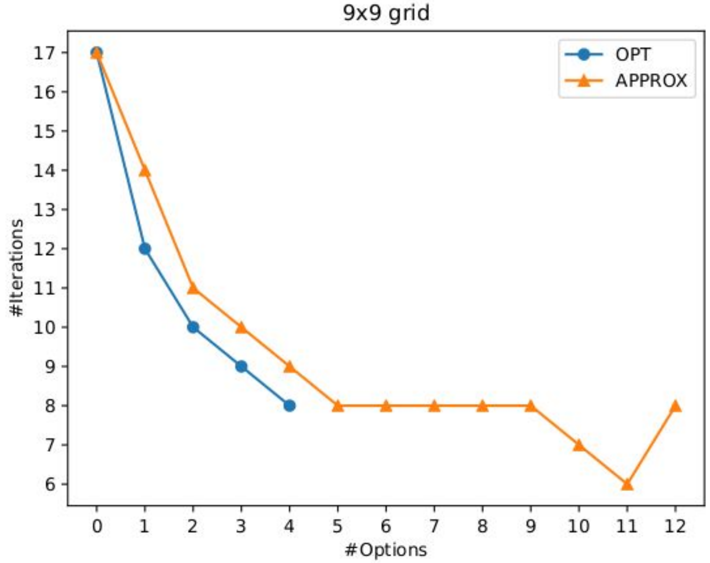
: Termination State: β(s)



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

	(a)	(b)	(c)	(d)	(e)	(f)
$L(O)$ =	11	16	8	11	10	12

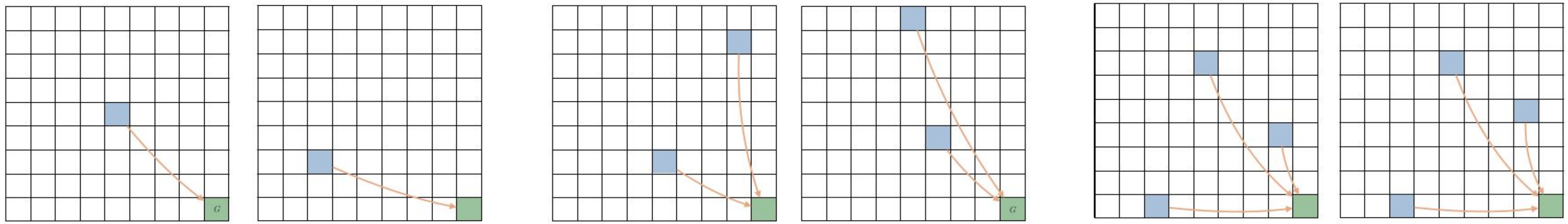
Evaluations: MIMO



MIMO: Minimize $L(O)$ subject to $|O| \leq k$

: Initiation State: *I*(s)

: Termination State: *β*(s)

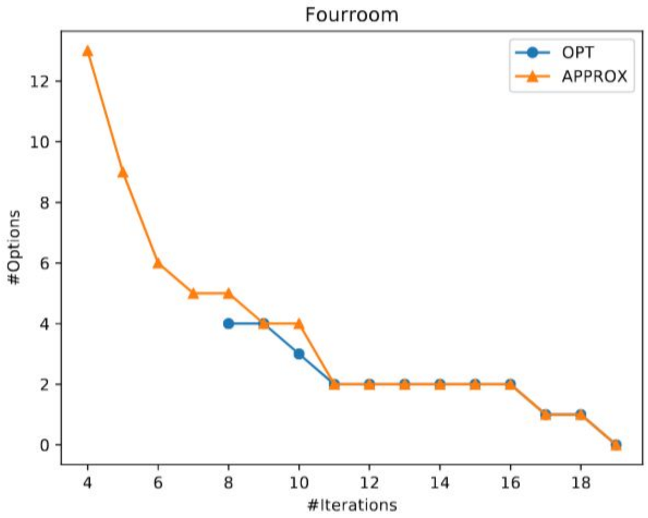


(a) optimal $k = 1$ | (d) approx. $k = 1$ | (b) optimal $k = 2$ | (e) approx. $k = 2$ | (c) optimal $k = 3$ | (f) approx. $k = 3$

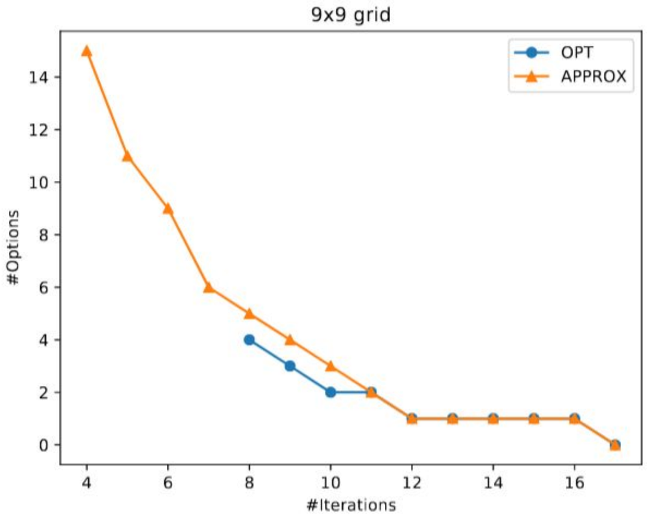
$L(O)$ = 12 14 10 11 9 10

Evaluations: MOMI

MOMI: Minimize $|O|$ subject to $L(O) \leqq L$



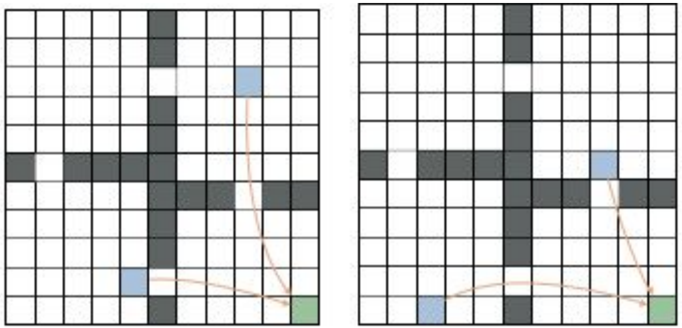
(c) Four Room (MOMI)



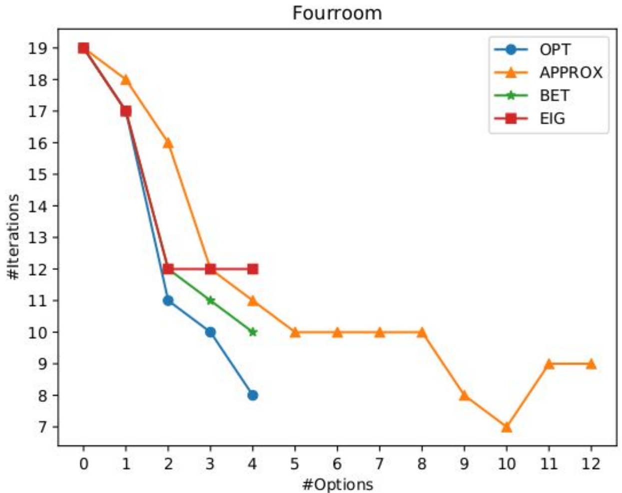
(d) 9×9 grid (MOMI)

Summary

1. Finding optimal behavioral abstraction is NP-hard
2. Provided polynomial time approximation algorithms which have a suboptimality bound if the MDP is deterministic
3. Empirically evaluated heuristically generated options



(a) optimal $k = 2$ (b) approx. $k = 2$

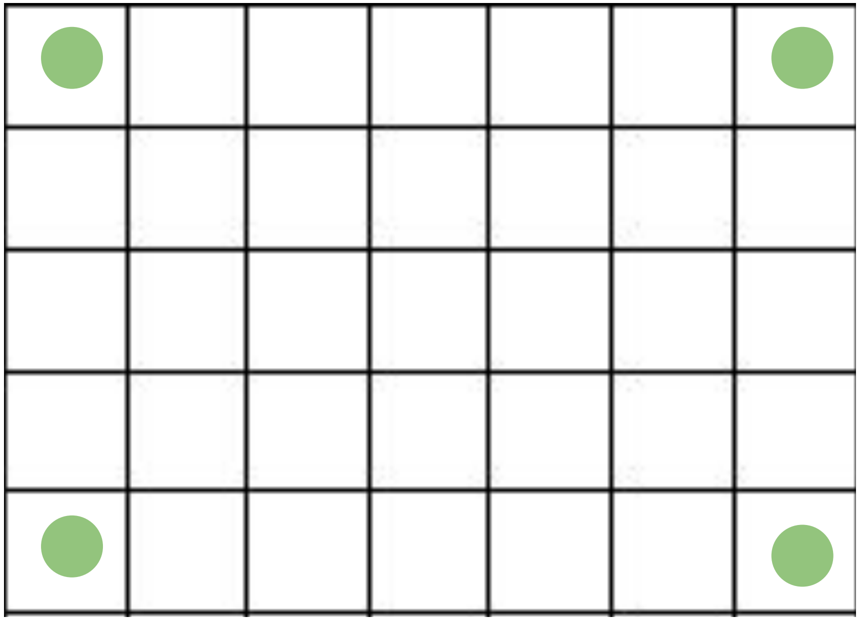


: Initiation State: I(s)

: Termination State: β(s)

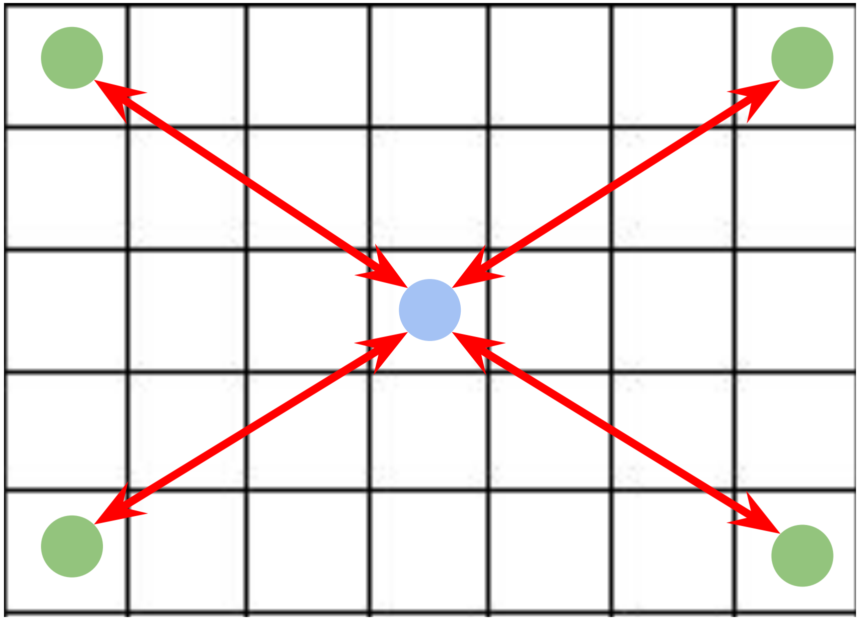
Future Work

Approximation algorithm for multitask setting

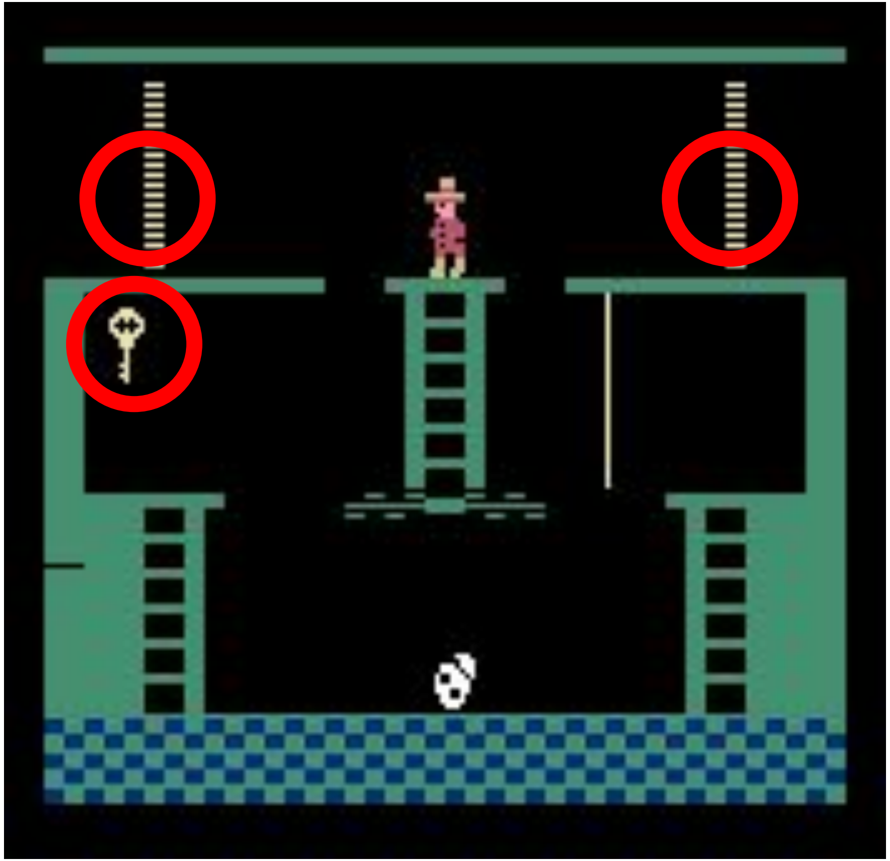


Future Work

Approximation algorithm for multitask setting



Future Work



Future Work