REINFORCEMENT LEARNING: THEORY AND PRACTICE

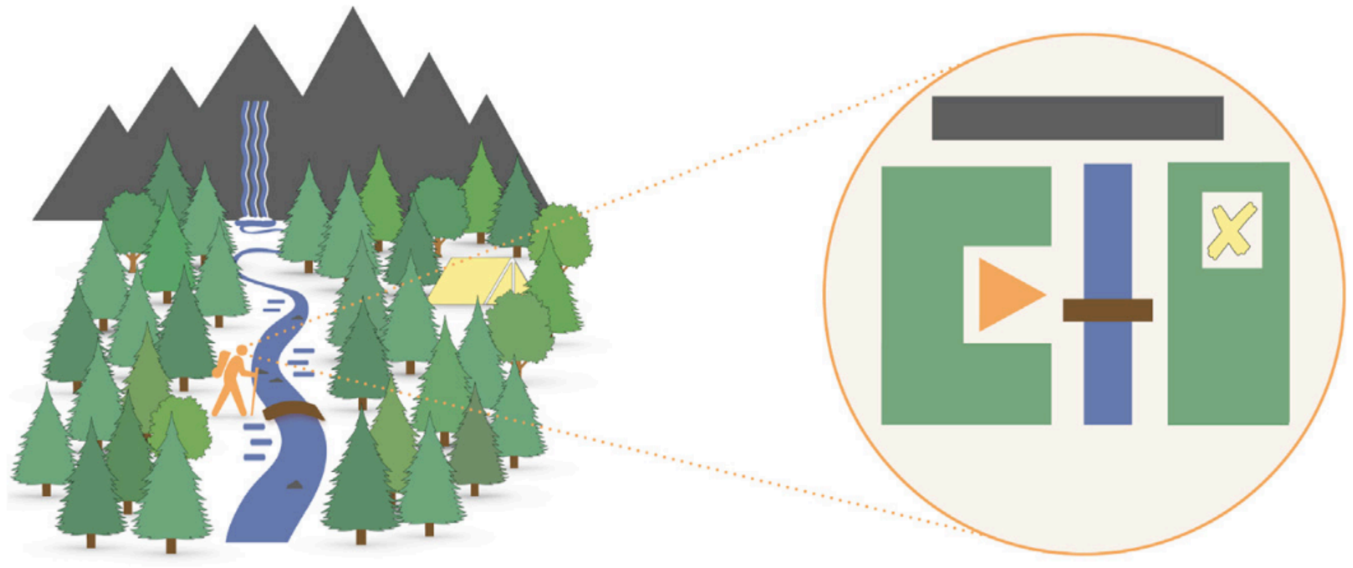
Abstractions and Options

Profs. Amy Zhang and Peter Stone

TEXAS
The University of Texas at Austin

Logistics Questions?

State abstractions



The Value of Abstraction Ho, M., Abel, D., Griffiths, T., and Littman, M.

What are abstractions and what are they good for?

What are abstractions?

Drop irrelevant information

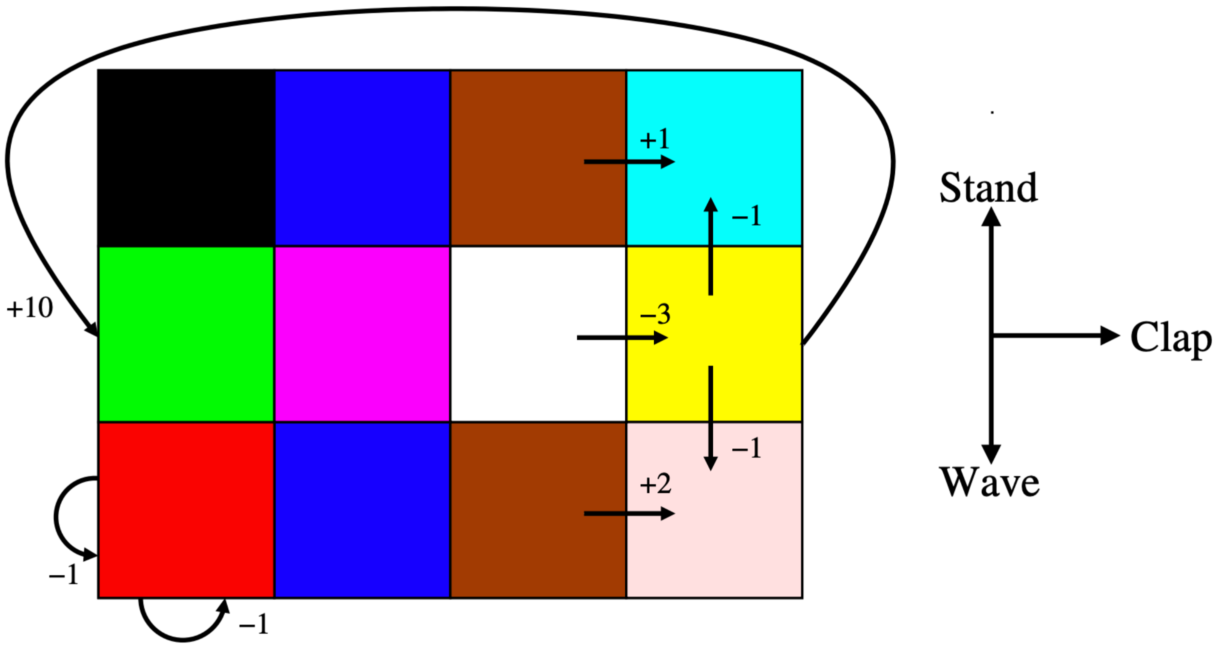
Allow for transfer and generalization

State abstractions have been studied as a way to distinguish relevant from irrelevant information in order to create a more compact representation for easier decision making and planning.

Policy Irrelevance Abstraction

Two states are considered equivalent if the optimal action from both states are the same.

$\gamma = 1$. Episodic case.
What states get grouped together?

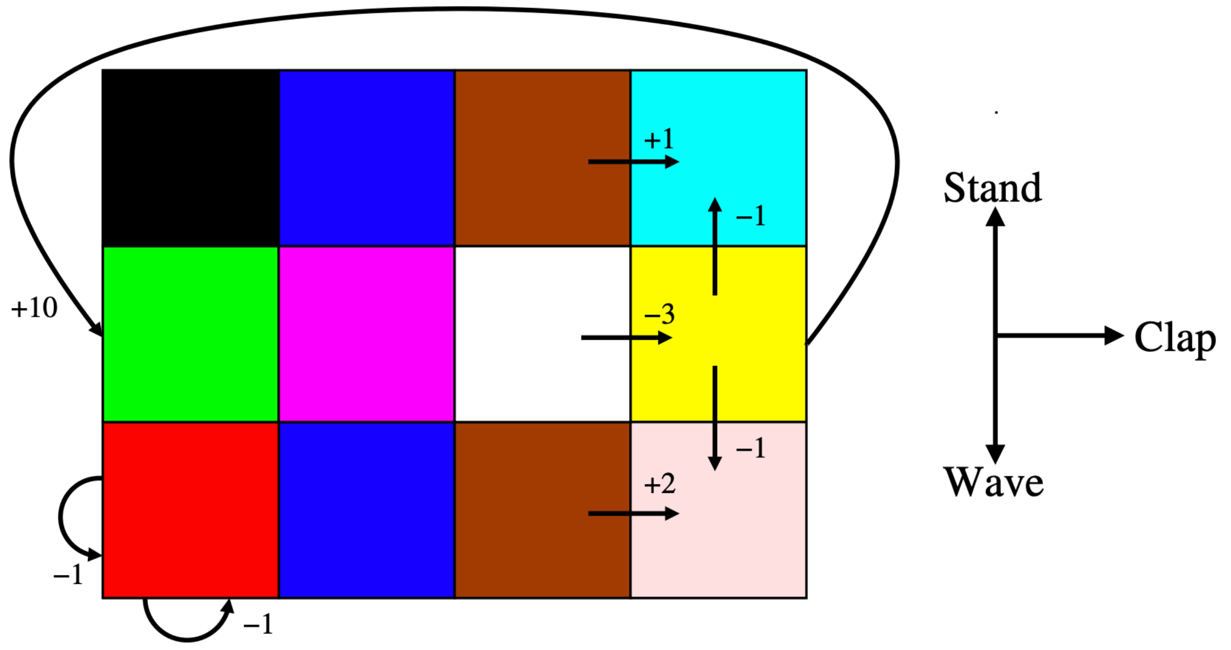


Policy Irrelevance Abstraction

Two states are considered equivalent if the optimal action from both states are the same.

Hold on! Important question: which optimal policy?

$\gamma = 1$. Episodic case.
What states get grouped together?

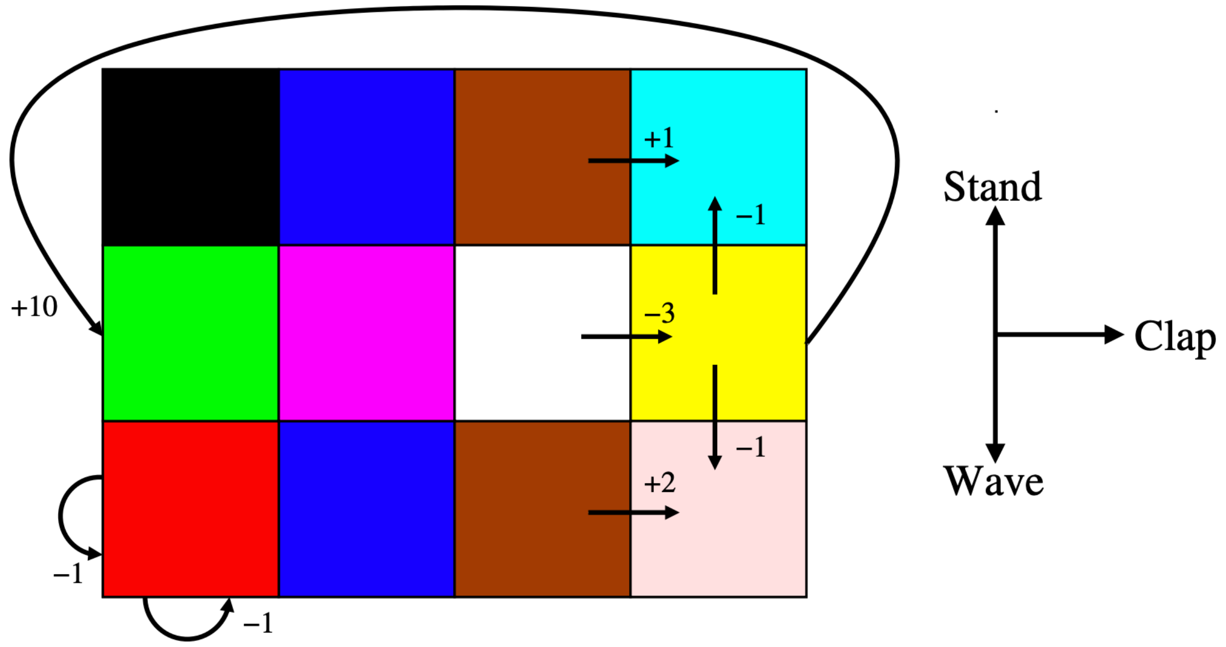


Policy Irrelevance Abstraction

Two states are considered equivalent if the optimal action from both states are the same.

Hold on! Important question: which optimal policy?

$\gamma = 1$. Episodic case. Optimal policy that also maximizes entropy. What states get grouped together?

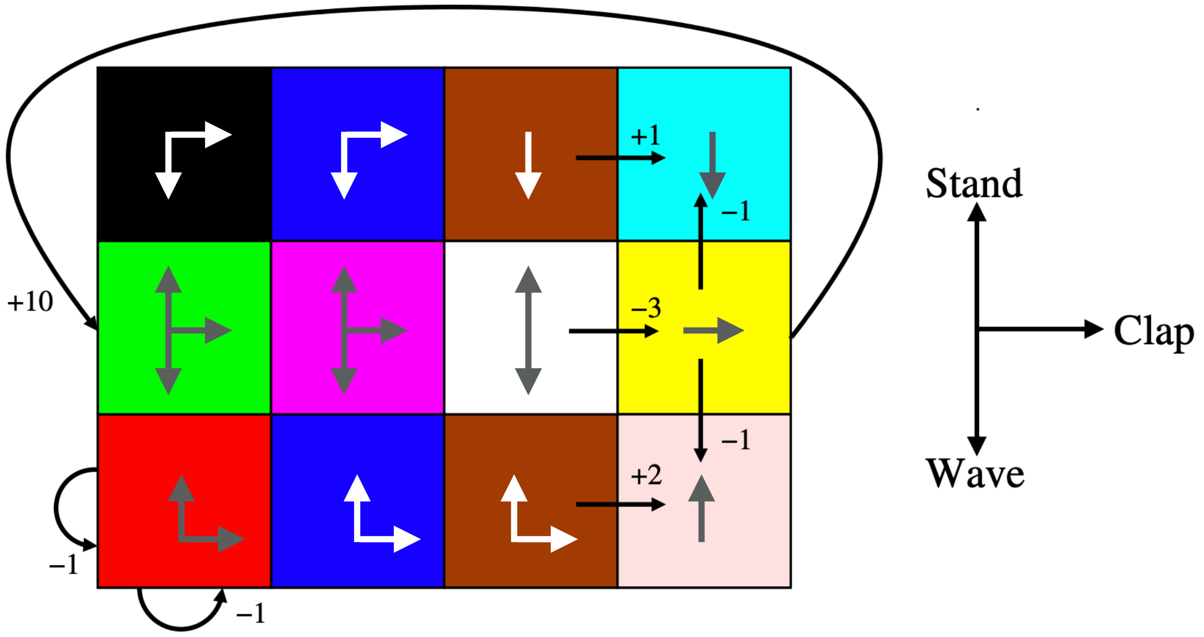


Policy Irrelevance Abstraction

Two states are considered equivalent if the optimal action from both states are the same.

Hold on! Important question: which optimal policy?

$\gamma = 1$. Episodic case. Optimal policy that also maximizes entropy. What states get grouped together?

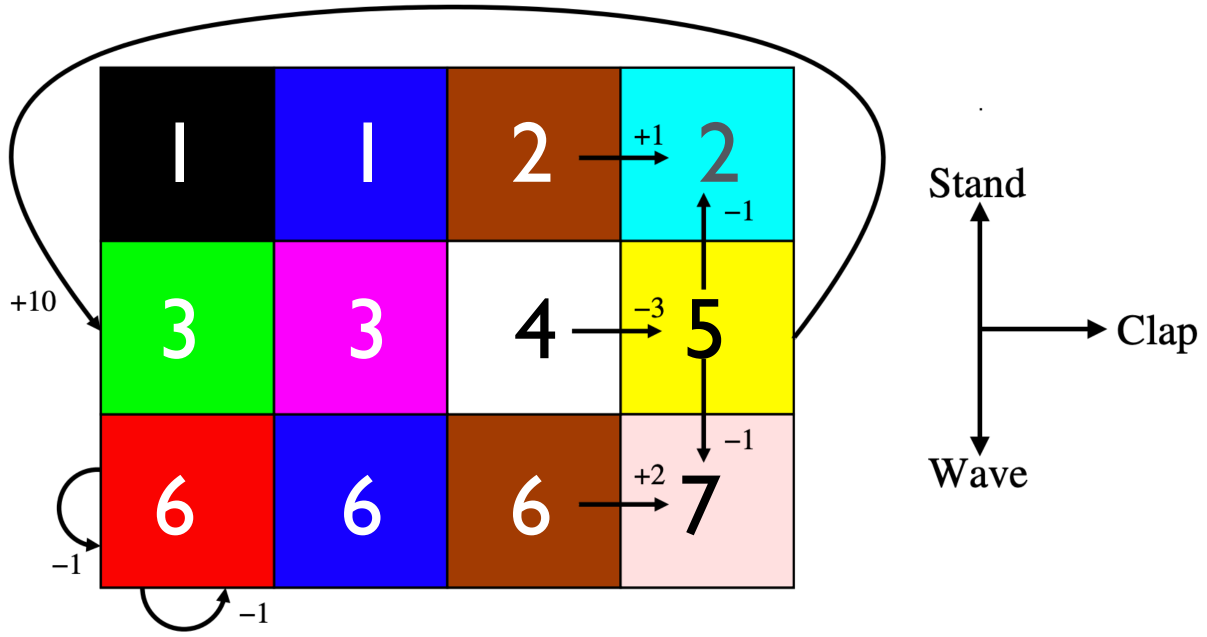


Policy Irrelevance Abstraction

Two states are considered equivalent if the optimal action from both states are the same.

Hold on! Important question: which optimal policy?

$\gamma = 1$. Episodic case. Optimal policy that also maximizes entropy. What states get grouped together?

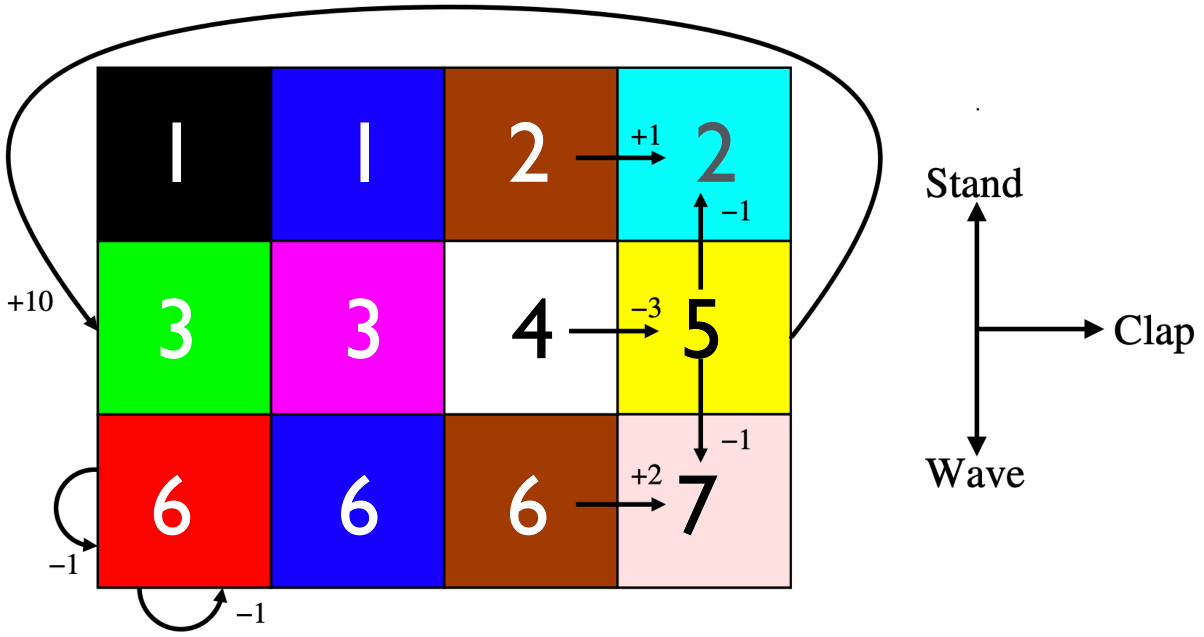


Policy Irrelevance Abstraction

Two states are considered equivalent if the optimal action from both states are the same.

$\gamma = 1$. Episodic case. Optimal policy that also maximizes entropy. What states get grouped together?

Can we learn the optimal policy from this abstraction?



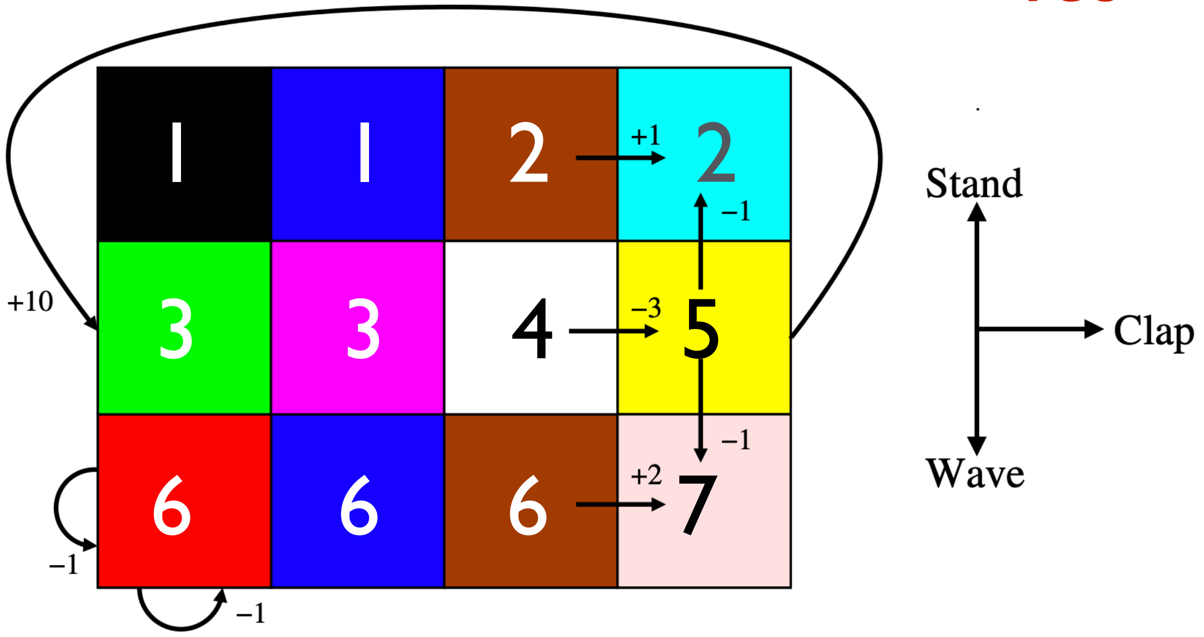
Policy Irrelevance Abstraction

Two states are considered equivalent if the optimal action from both states are the same.

$\gamma = 1$. Episodic case. Optimal policy that also maximizes entropy. What states get grouped together?

Yes

Can we learn the optimal policy from this abstraction?

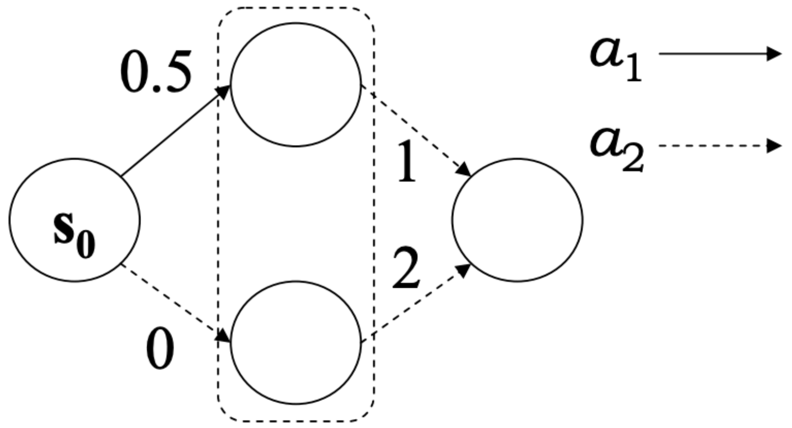


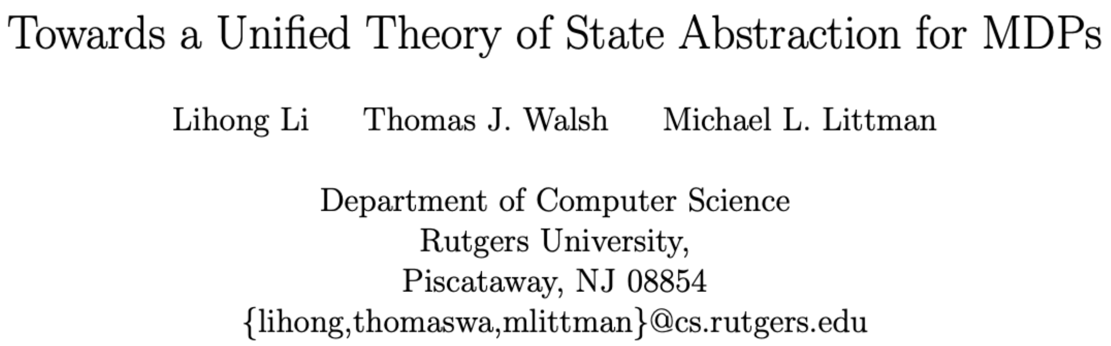
Can we always?

Arnav Jain
What are some scenarios where abstraction leads to no optimal solution/the non-optimal solution? How can we ensure that this doesn’t happen in our abstraction decisions?

Can we always?

Arnav Jain
What are some scenarios where abstraction leads to no optimal solution/the non-optimal solution? How can we ensure that this doesn't happen in our abstraction decisions?





Towards a Unified Theory of State Abstraction for MDPs

Lihong Li Thomas J. Walsh Michael L. Littman

Department of Computer Science
Rutgers University,
Piscataway, NJ 08854
{lihong,thomaswa,mlittman}@cs.rutgers.edu

A stricter form of abstraction: Bisimulation

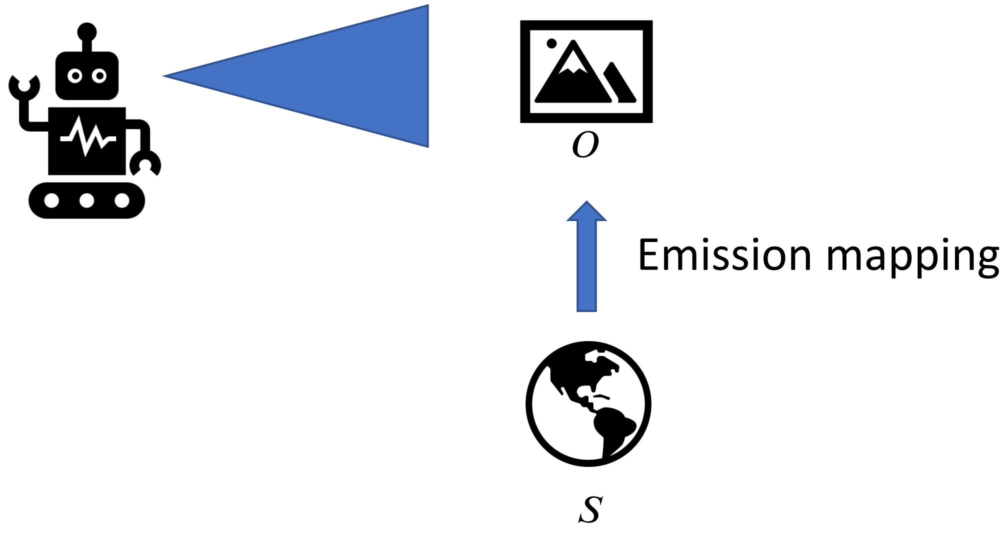
Definition

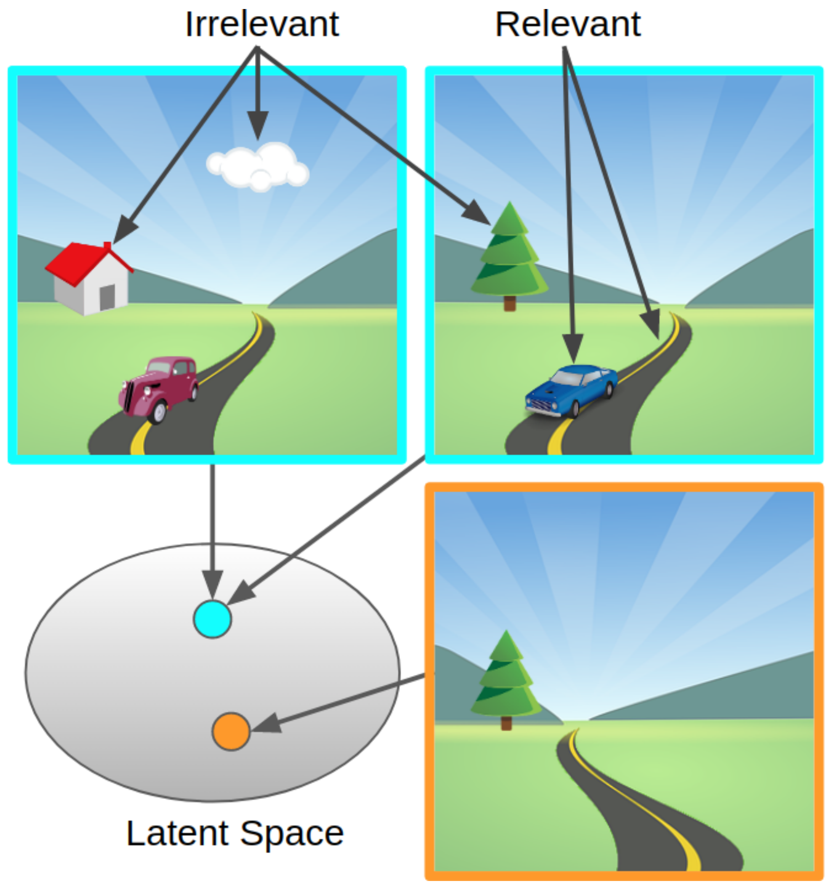
Given an MDP $\mathcal{M}$, an equivalence relation B between states is a bisimulation relation if, for all states $s_i, s_j \in \mathcal{S}$ that are equivalent under B (denoted $s_i \equiv_B s_j$) the following conditions hold:

$$\begin{aligned} \mathcal{R}(s_i, a) &= \mathcal{R}(s_j, a) \quad \forall a \in \mathcal{A}, & (1) \\ \mathcal{P}(G|s_i, a) &= \mathcal{P}(G|s_j, a) \quad \forall a \in \mathcal{A}, \quad \forall G \in \mathcal{S}_B, & (2) \end{aligned}$$

where $\mathcal{S}_B$ is the partition of $\mathcal{S}$ under the relation B (the set of all groups G of equivalent states), and $\mathcal{P}(G|s, a) = \sum_{s' \in G} \mathcal{P}(s'|s, a)$.

A realistic additional assumption



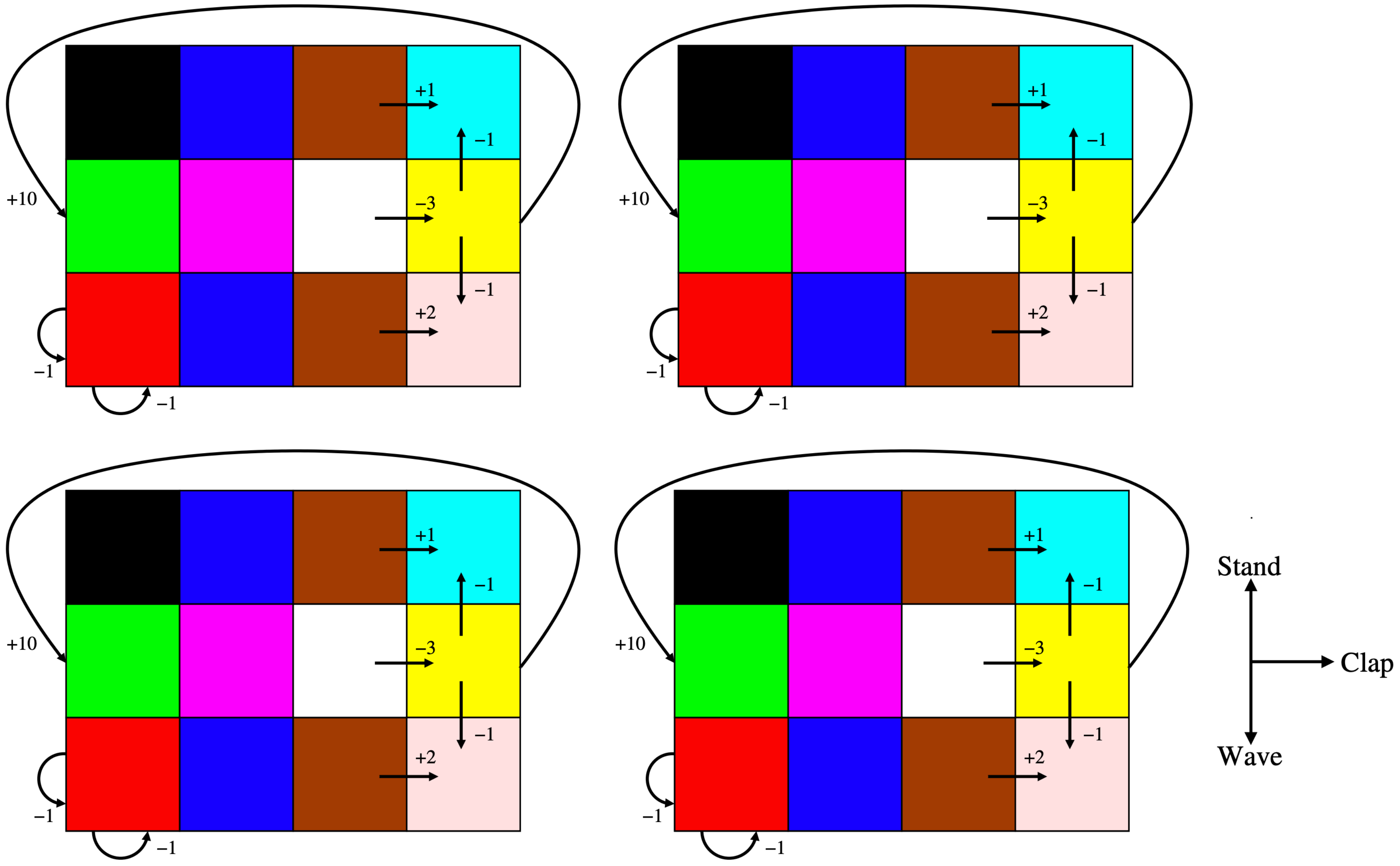


Goal: Generalization to new observations *where the underlying MDP is the same*

Solution: Ignore irrelevant information

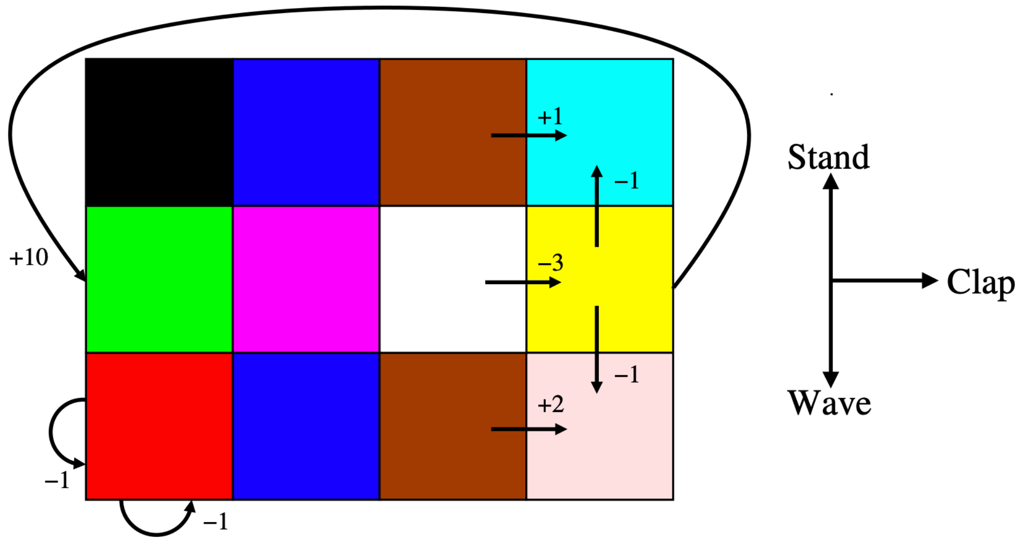
Exercise: State Abstractions

What's the minimal bisimulation abstraction for this MDP?

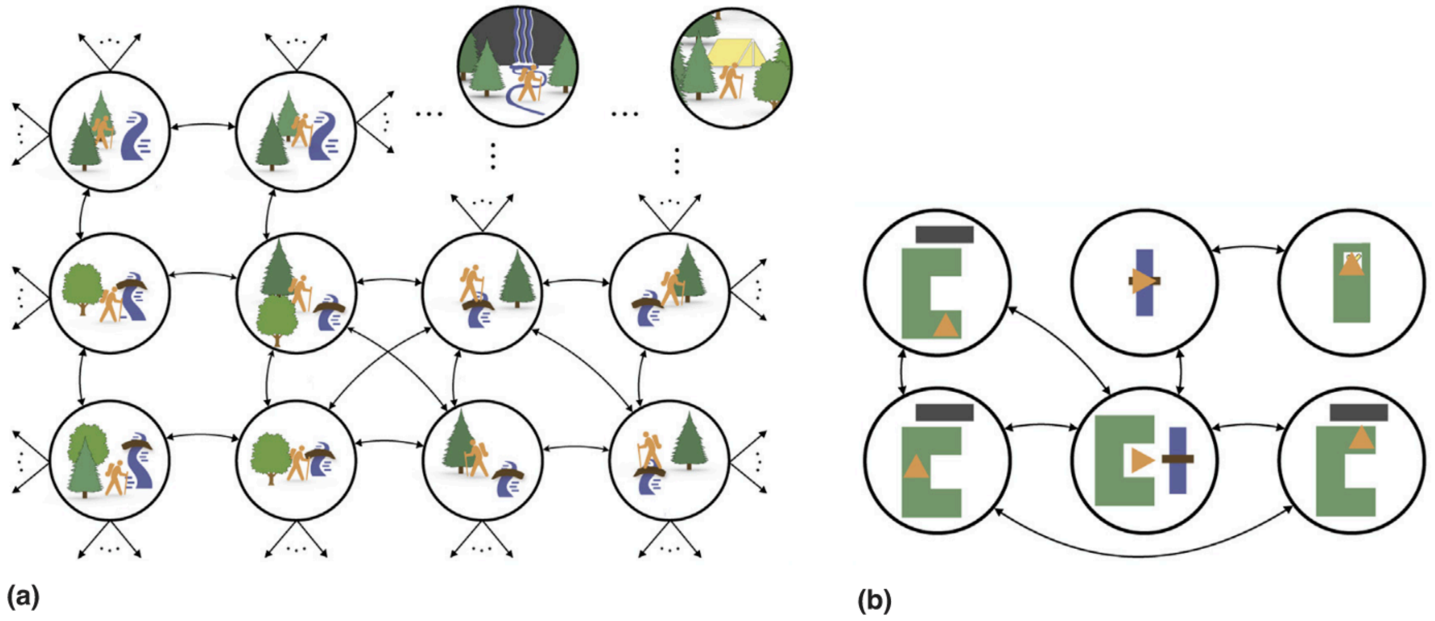


Exercise: State Abstractions

No matter which MDP we're in, the behavior of the corresponding state is exactly the same.



State + temporal abstractions

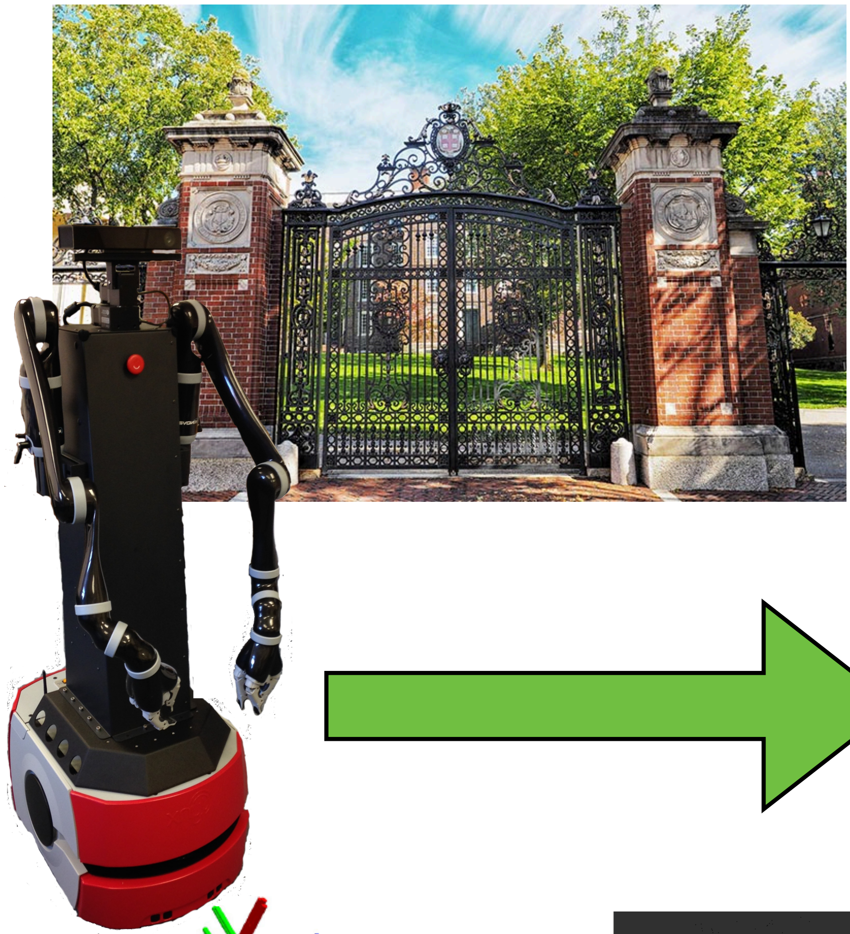

(a) (b)

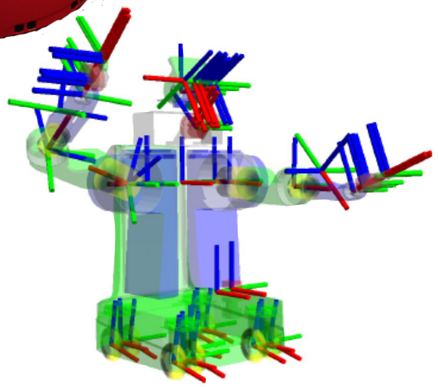
The Value of Abstraction Ho, M., Abel, D., Griffiths, T., and Littman, M.

Hierarchical Reinforcement Learning

George Konidaris
gdk@cs.brown.edu

Why Hierarchies?





Skill Hierarchies

Hierarchical RL: base hierarchical control on *skills*.

- Component of behavior.
- Performs continuous, low-level control.
- Can treat as discrete action.

Behavior is modular and compositional.

Skills are like *subroutines*.

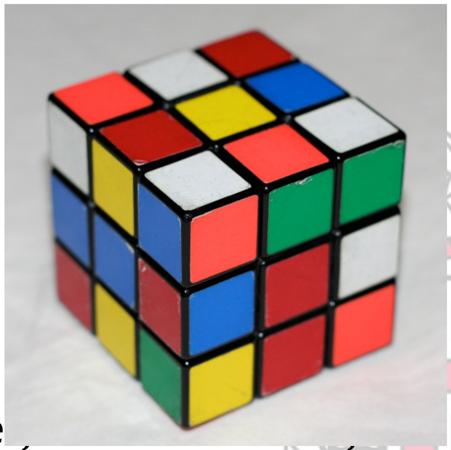
```
d
            ):
            x

    return -x
```

Development

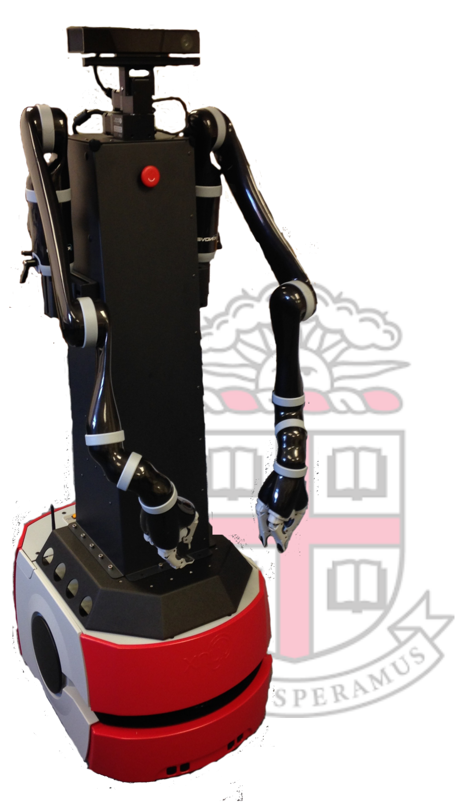
Specialization

[Wilke l]


Simplification

Forms of Abstraction

$$\begin{tikzcd}
\langle \bar{S}, \bar{A}, R, T, \gamma \rangle \\
\langle S, A, R, T, \gamma \rangle \arrow[u, "\text{state abstraction}", "\text{action abstraction}"']
\end{tikzcd}$$



The Options Framework

Options

Options Framework: theoretical basis for skill acquisition, learning and planning using higher-level actions (options).

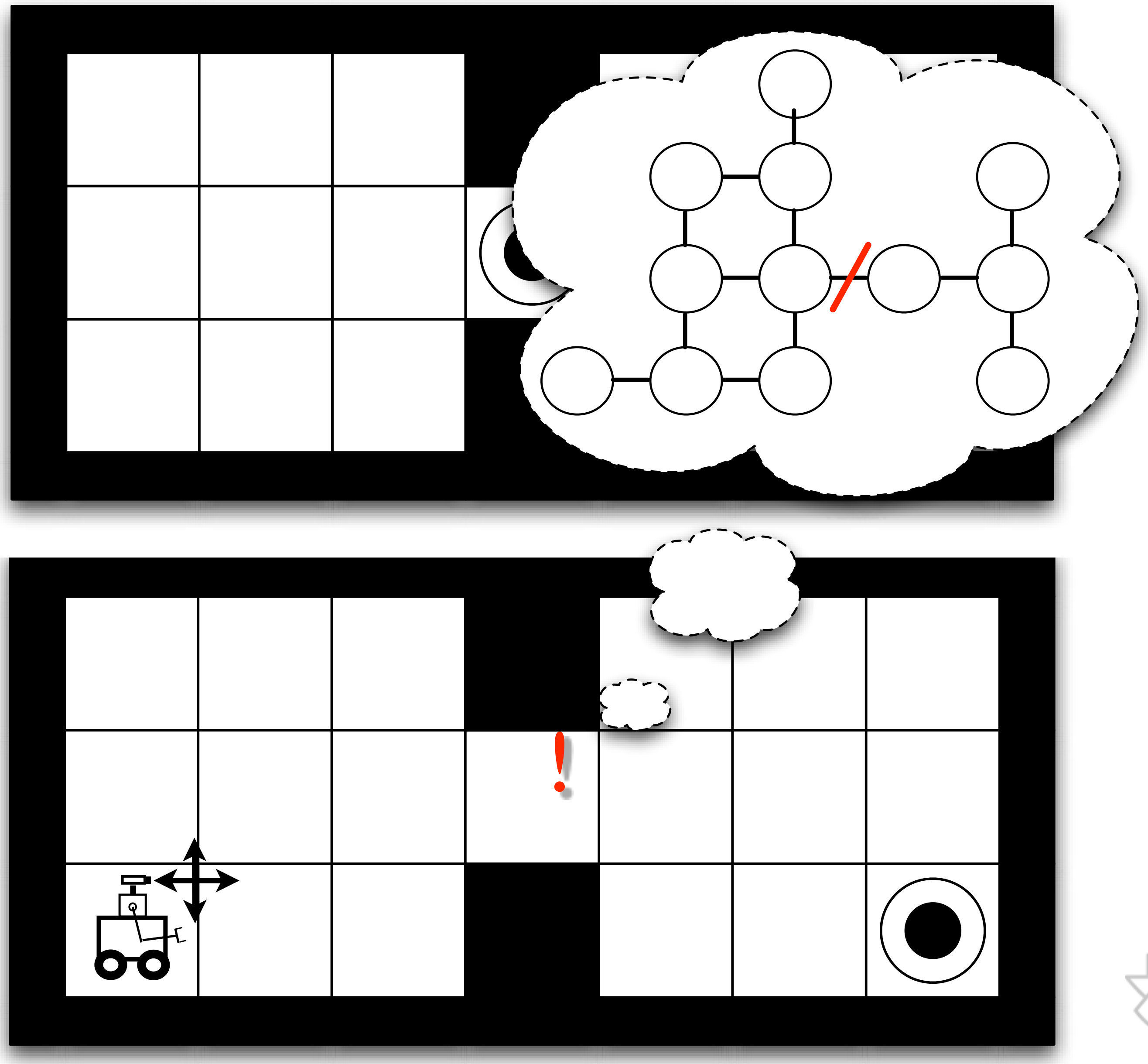
RL typically solves a *single* problem *monolithically*.

Action abstraction:

- Create and use higher-level macro-actions.
- Problem now contains subproblems.
- Each subproblem is also an RL problem.

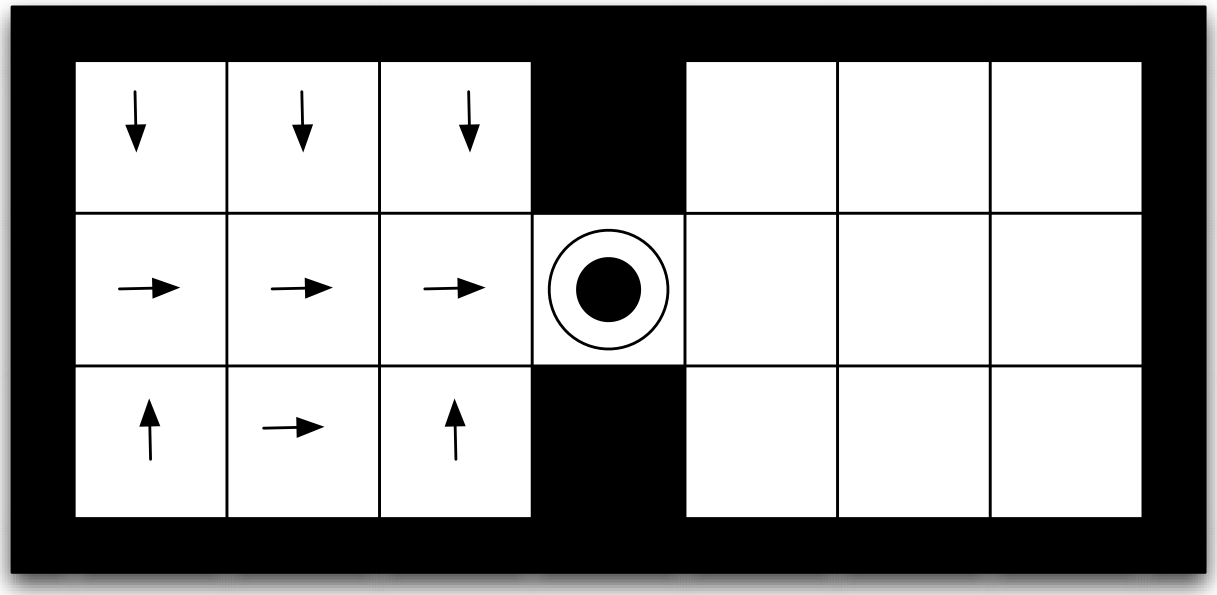
[Sutton, Precup, and Singh, 1999]

Hierarchical RL

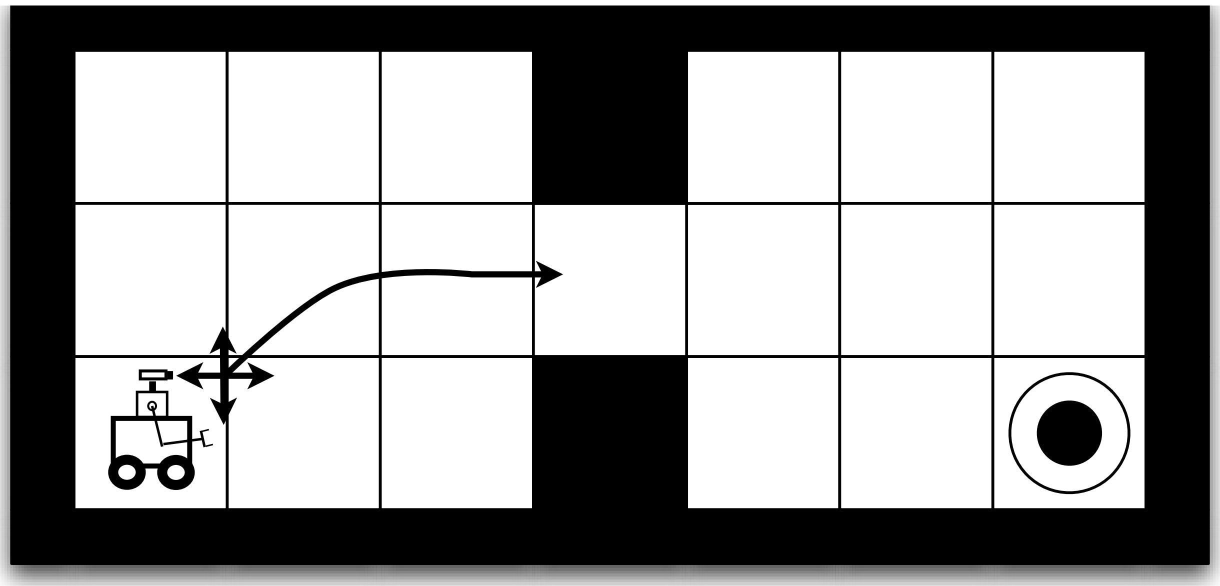


Hierarchical RL

Skill



Problem

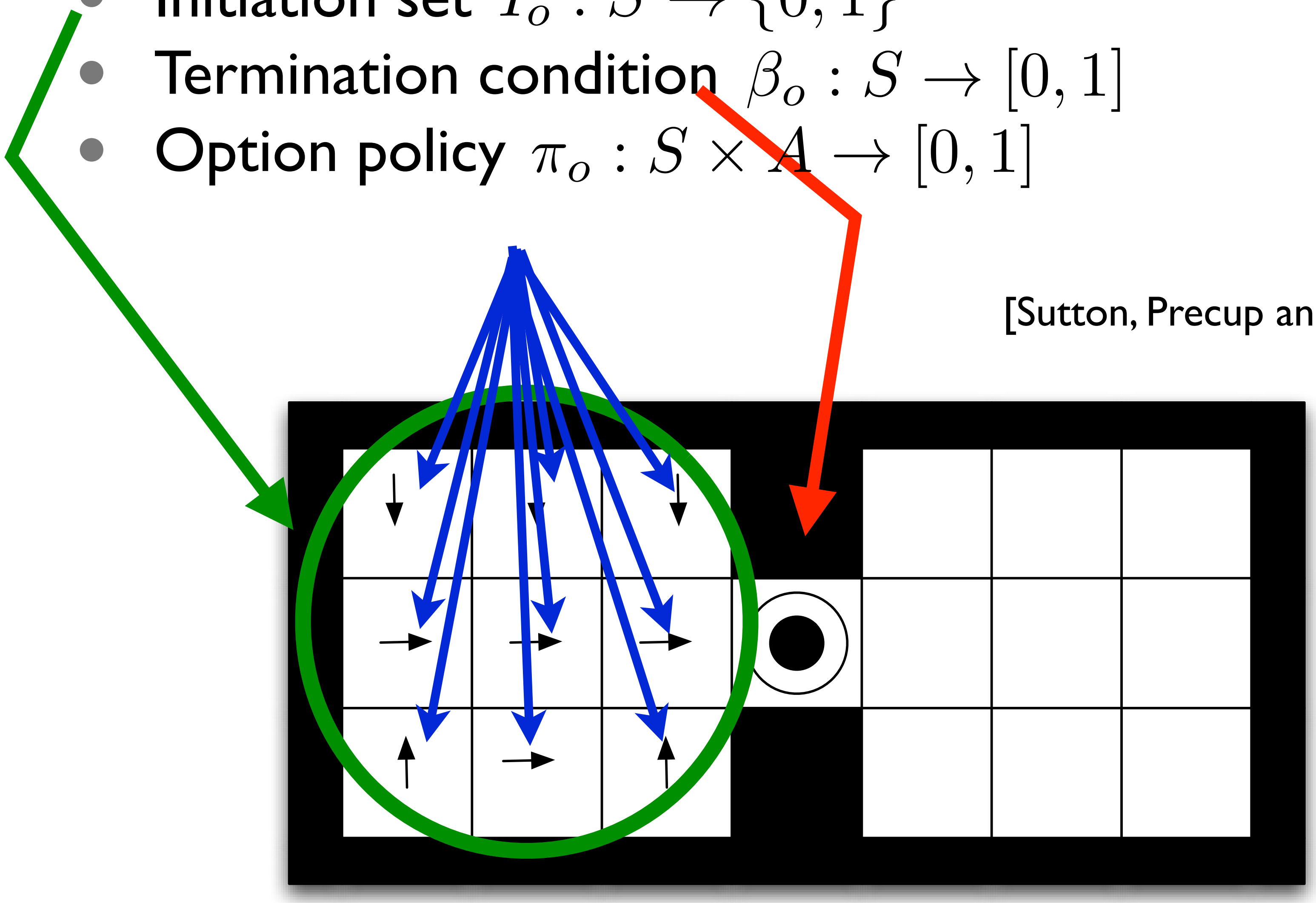


The Options Framework

An option is one formal model of a skill.

An option *o* is a policy unit:

- Initiation set $I_o : S \to \{0, 1\}$
- Termination condition $\beta_o : S \to [0, 1]$
- Option policy $\pi_o : S \times A \to [0, 1]$



[Sutton, Precup and Singh 1999]

Actions as Options

A primitive action *a* can be represented by an option:

- $I_a(s) = 1, \forall s \in S$
- $\beta_a(s) = 1, \forall s \in S$
- $\pi_a(s,b) = \begin{cases} 1 & a = b \\ 0 & \text{otherwise} \end{cases}$

A primitive action can be executed anywhere, lasts exactly one time step, and always chooses action *a*.

Questions

Given an MDP:

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

... let's replace A with a set of options O (some of which may be primitive actions).

- How do we characterize the resulting problem?
- How do we plan using options?
- How do we learn using options?
- How do we characterize the resulting policies?

SMDPs

The resulting problem is a *Semi-(Markov Decision Process)*. This consists of:

S	Set of states
O	Set of options
$P(s', t \mid o, s)$	Transition model
$R(s', s, t)$	Reward function
γ	Discount factor (per step)

In this case:

- All times are natural numbers.
- "Semi" here means transitions can last *t* timesteps.
- Transition and reward function involve time taken for option to execute.

Planning?

option model

Easy

$$Q^\pi(s,o) = \mathbb{E}_{t,s'}[R(s',s,t)] + \mathbb{E}_{t,s'}[\gamma^t \pi(s',o')Q^\pi(s',o')]$$

where

$$\mathbb{E}_{t,s'}[R(s',s,t)] = \sum_{t,s'} \boxed{P(s',t|o,s)}\boxed{R(s',s,t)}$$

$$\mathbb{E}_{t,s'}[\gamma^t \pi(s',o')Q^\pi(s',o')] = \sum_{t,s'} P(s',t|o,s)\gamma^t \pi(s',o')Q^\pi(s',o')$$

All things flow from Bellman.

Learning and Planning

$$Q^\pi(s,o) = \mathbb{E}_{t,s'}[R(s',s,t)] + \mathbb{E}_{t,s'}[\gamma^t \pi(s',o') Q^\pi(s',o')]$$

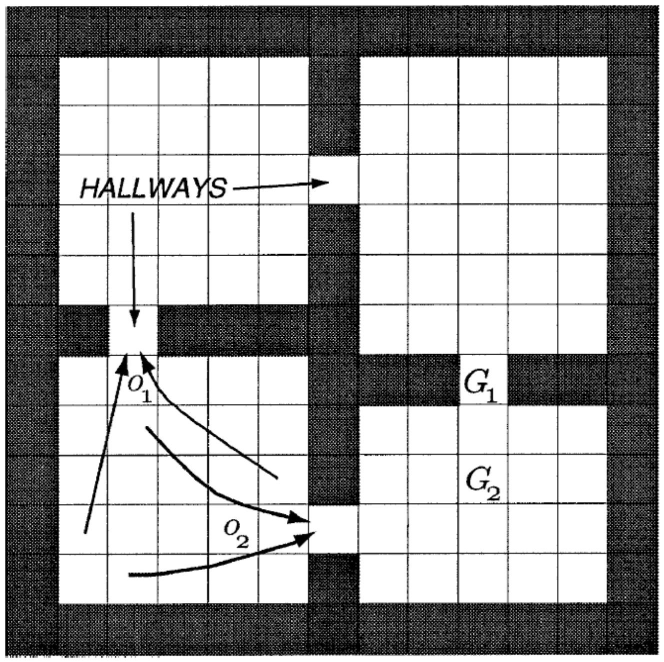
For learning:

- Stochastic samples.
- Use SMDP Bellman equation.

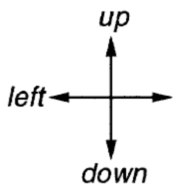
For planning:

- Synchronous Value Iteration via SMDP Bellman eqn

Example



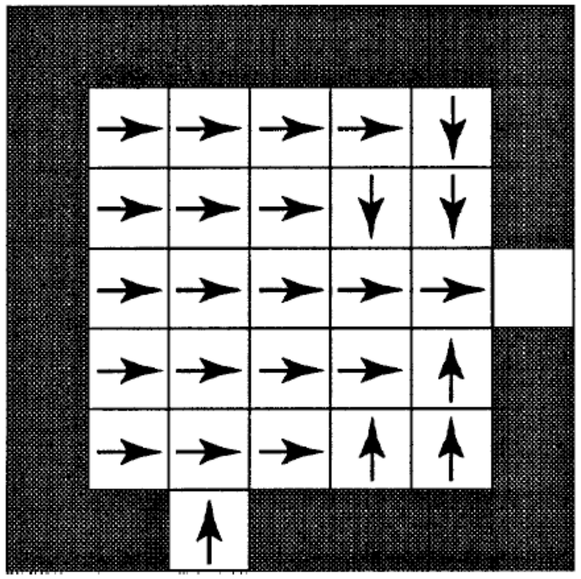
4 stochastic primitive actions



Fail 33% of the time

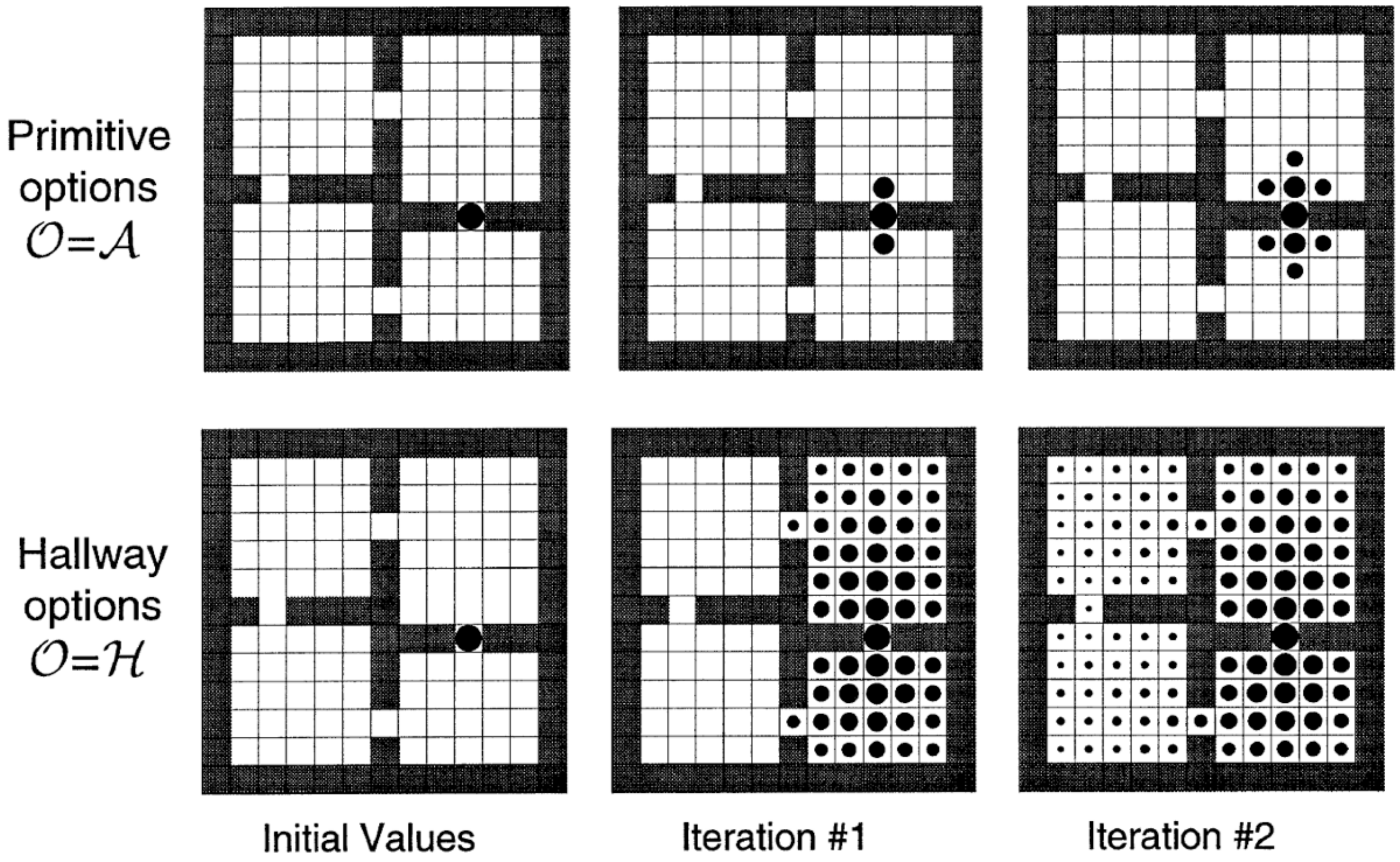
8 multi-step options

(to each room's 2 hallways)



(Sutton, Precup and Singh, AIJ 1999)

Example

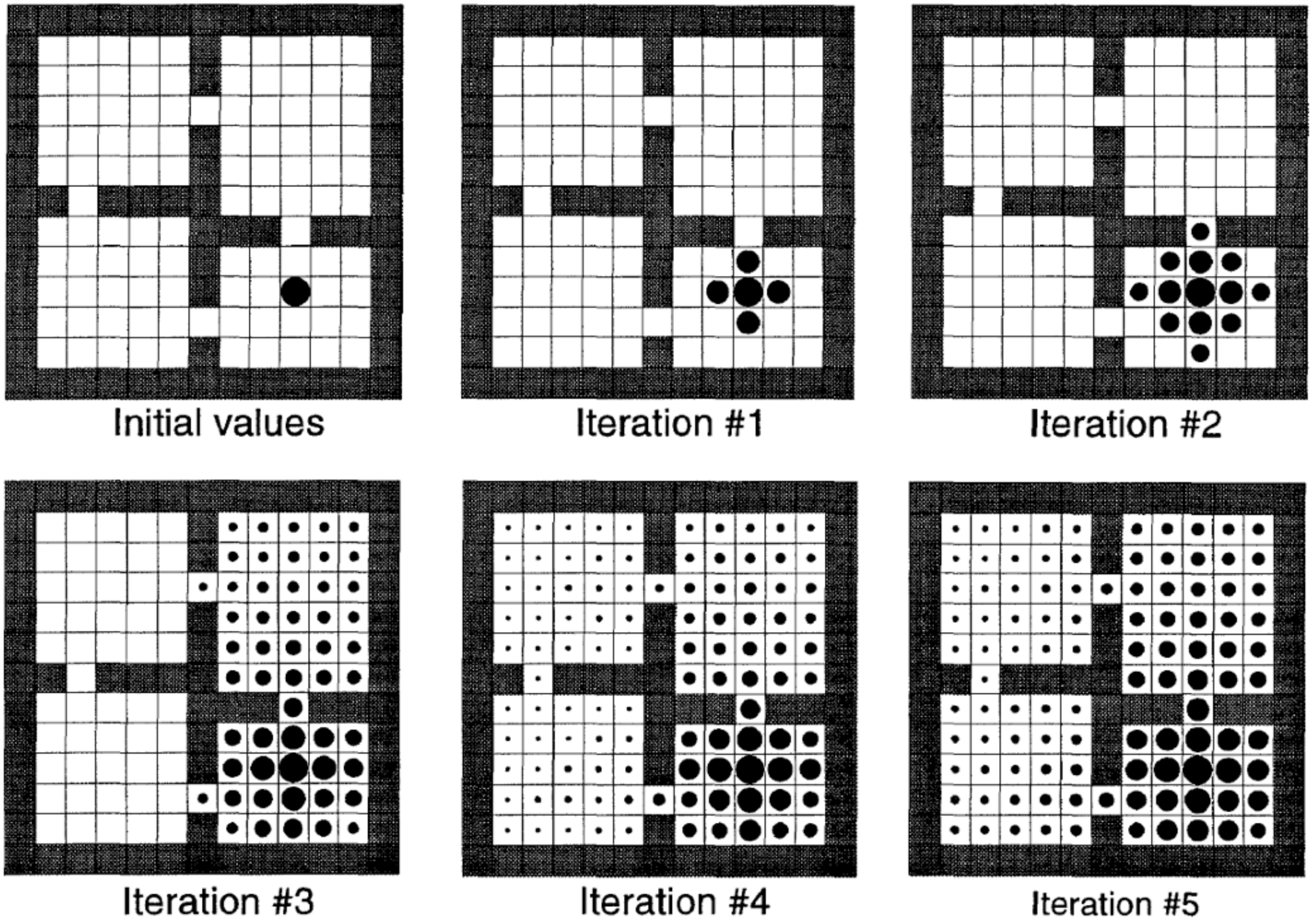


(Sutton, Precup and Singh, AIJ 1999)

Example

Primitive and hallway options

$\mathcal{O}=\mathcal{A}\cup\mathcal{H}$



(Sutton, Precup and Singh, AIJ 1999)

Final note: policies.

A policy over an MDP with primitive actions is a *Markov policy*:

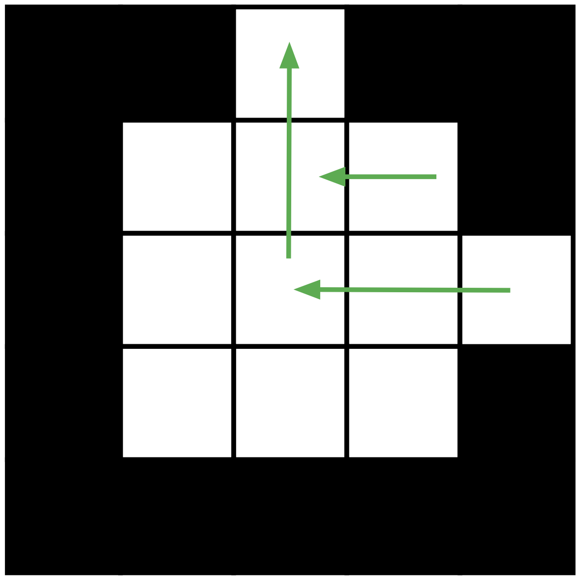
$$\pi : S \times A \rightarrow [0, 1]$$

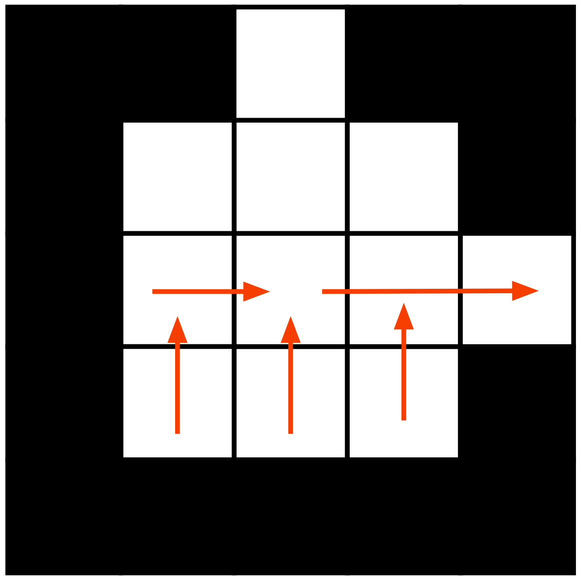
A policy over an MDP with options could also be Markov:

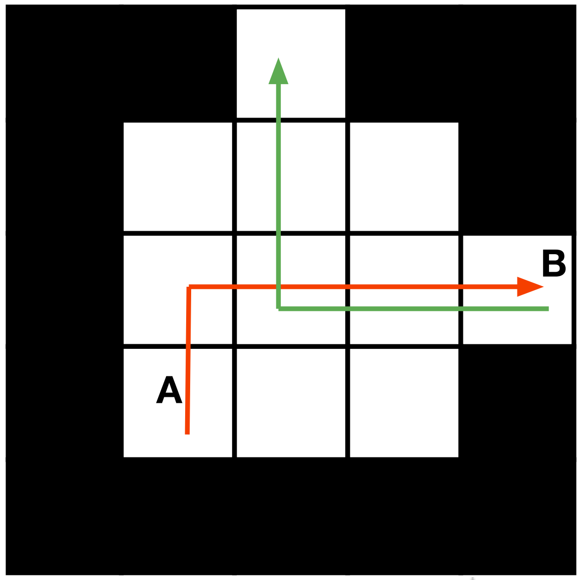
$$\pi : S \times O \rightarrow [0, 1]$$

... but this could imply a policy in the original MDP that is not, because the probability of taking an action at a state *depends on the option currently running*.

Example


Option A


Option B



Policy

So

A Markov policy for an SMDP may result in a *semi-Markov* policy for the underlying MDP.

(Even if the options are Markov options!)

Here, semi-Markov means that the probability of taking a primitive action at each step depends on more than the current state.

What are Options For?

What are Options For?

Lots of things!

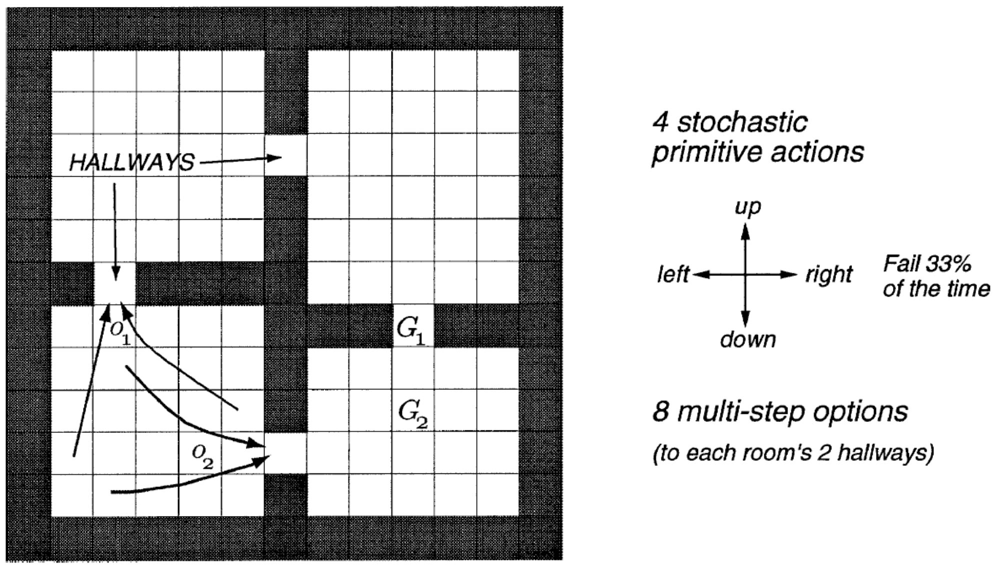
A few salient points:

- Rewiring.
- Transfer.
- Skill-Specific State Abstractions.

Rewiring

Adding an option changes the connectivity of the MDP.
This affects:

- Learning and planning.
- Exploration.
- State-visit distribution.
- *Diameter of problem.*



(Sutton, Precup and Singh, AIJ 1999)

Transfer

Use experience gained while solving one problem to improve performance in another.

Skill transfer:

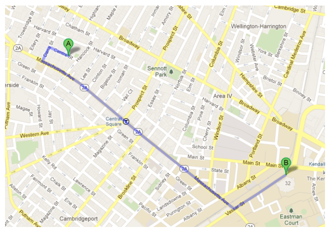
- Use options as mechanism for transfer.
- Transfer *components* of solution.
- Can drastically improve performance
- ... even if it takes a lot of effort to learn them.

General principle: **subtasks recur.**

Skill-Specific Abstractions

Options provide opportunities for abstraction

- Split high-dimensional problem into subproblems ...
- ... such that each one supports a solution using an abstraction.



Working hypothesis: *behavior is modular and compositional* ***and piecewise low-dimensional.***

Skill Acquisition

Skill Discovery

Where do skills come from?

Research goal: discover options autonomously, through interaction with an environment.

- Typically *subgoal options*.
- This means that we must determine β_o.
- Sometimes also R_o.

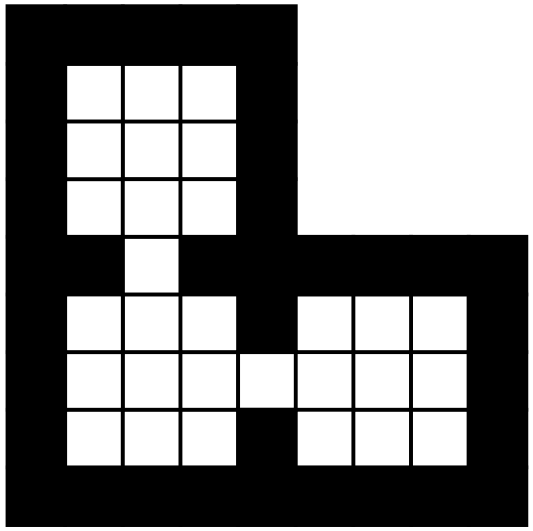
The question then becomes:

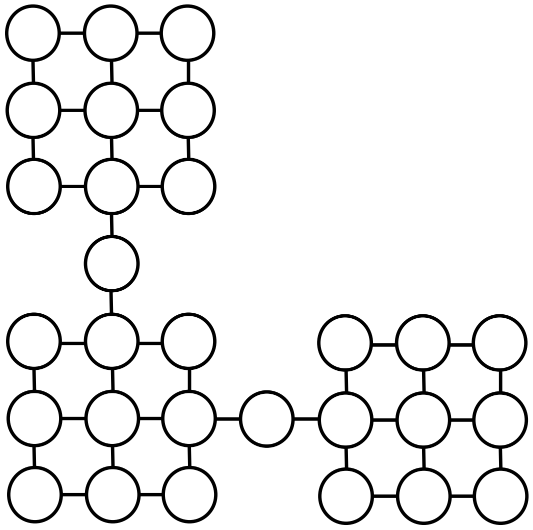
- Which states are good subgoals?

Betweenness Centrality

Consider an MDP as a graph.

- States are vertices.
- Edges indicate possible transition between two states.





Further, let us assume a task distribution over start states and goal pairs:

- $P_T(s, e)$

(Simsek and Barto, 2008)

Betweenness Centrality

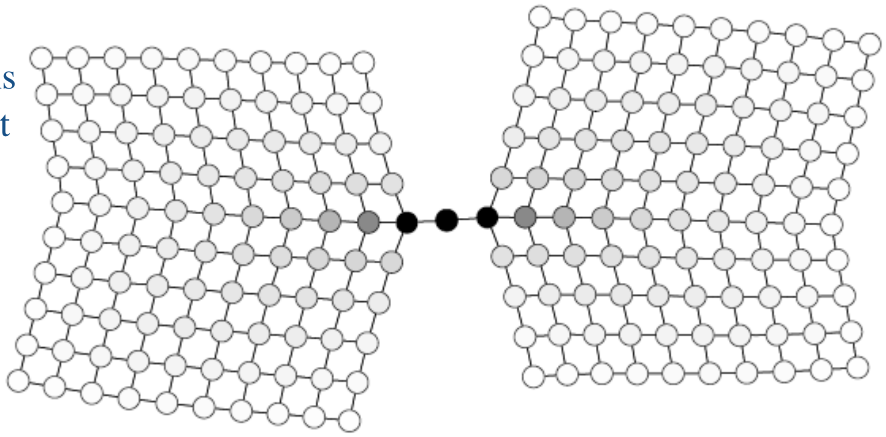
We can define the *betweenness centrality* of a vertex (state) as:

number of such paths that pass through vertex v

$$\sum_{s,e} \frac{\sigma_{se}(v)}{\sigma_{se}} w_{se}$$

weight assigned to paths from vertex s to vertex t

number of shortest paths from vertex s to vertex e



Bottleneck concept!

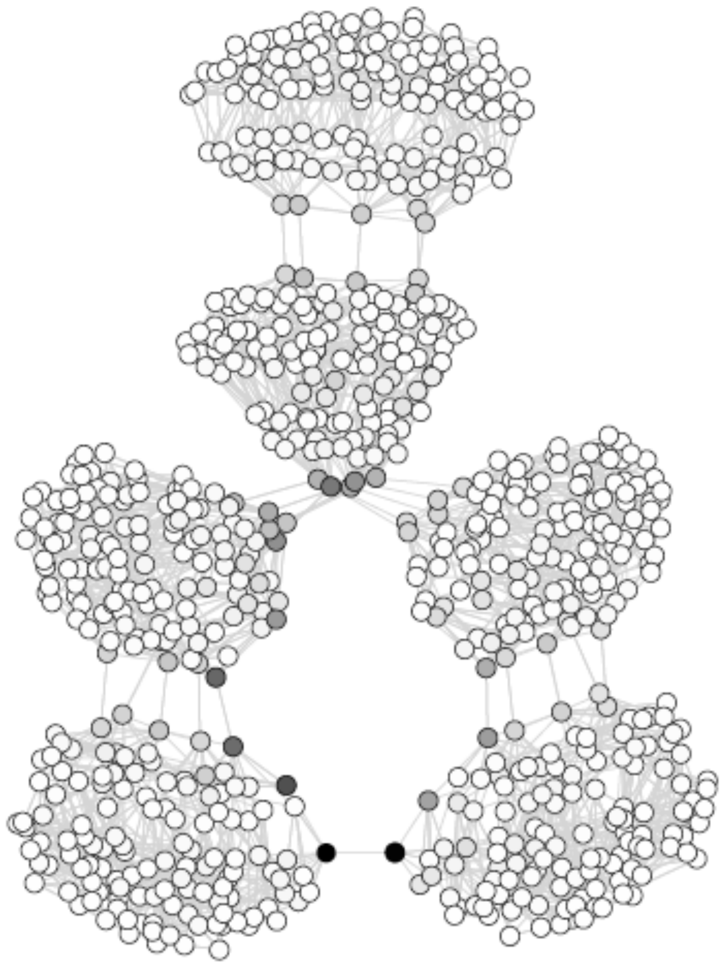
This indicates it probability of being on a shortest path from s to e; if we define:

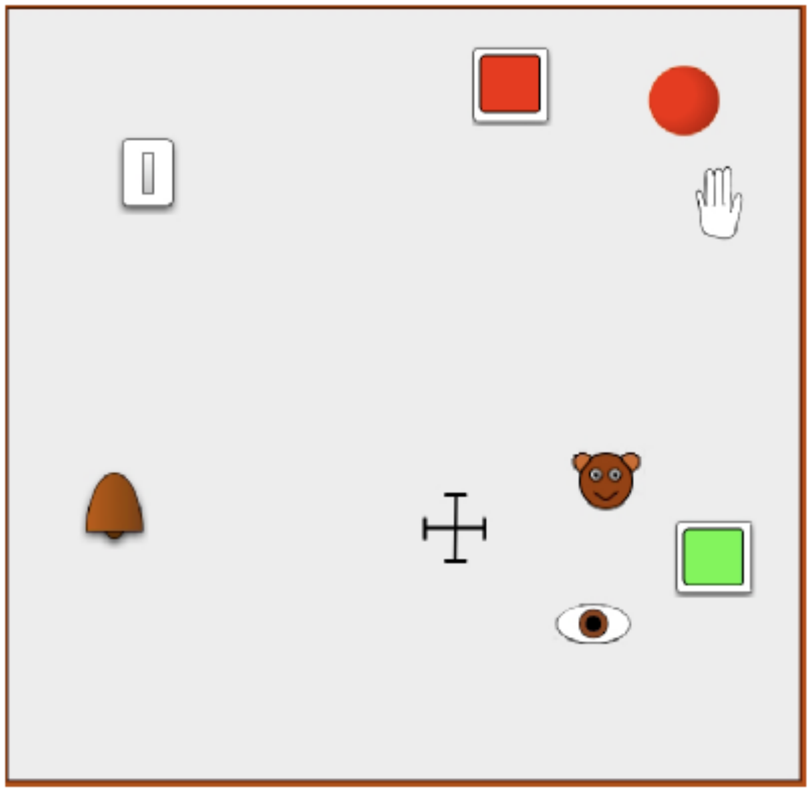
- *Shortest path* as *optimal solution.*
- $w_{se} = P_T(s, e)$

... then we get something sensible for RL.

(Simsek and Barto, 2008)

Betweenness Centrality





(Simsek and Barto, 2008)

Covering Options

More modern:

- Formulate a specific objective
- Find options with formal link to objective

E.g., finding options to aid with exploration:

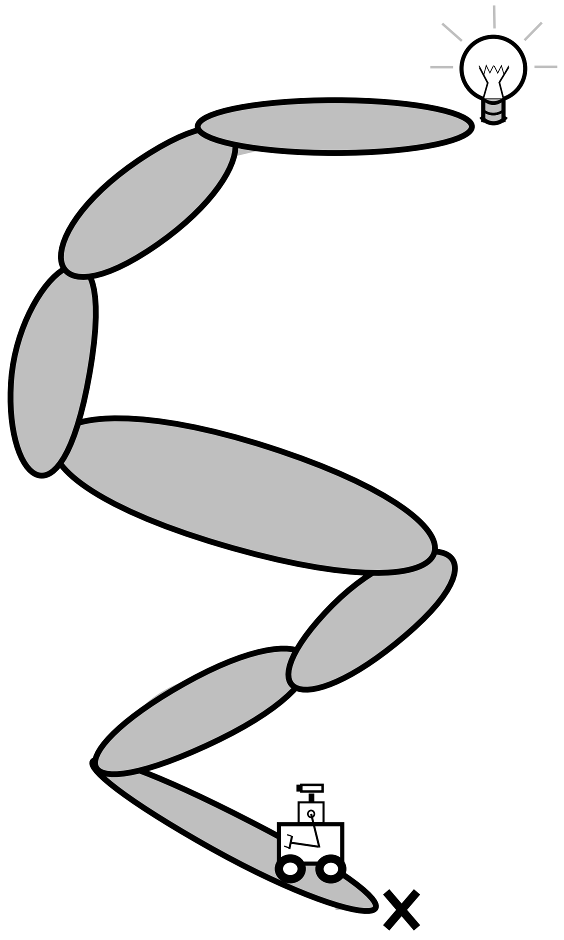
- *"the difficulty of discovering a distant rewarding state in an MDP is bounded by the expected cover time of a random walk over the graph induced by the MDP's transition dynamics"*

- Therefore, find options to minimize cover time.
- This is NP-Hard.
- Bounded-suboptimal approximation algorithm.

[Jinnai et al., 2019]

What About Continuous Domains?

Skill Chaining



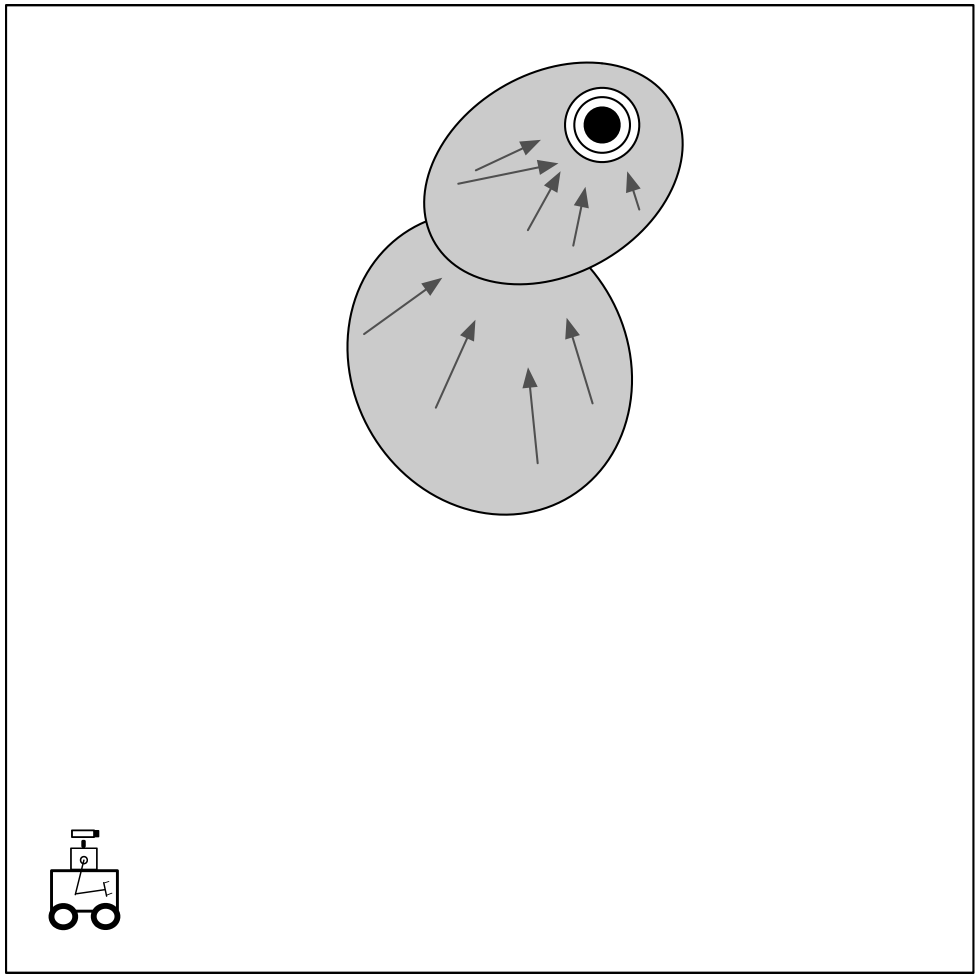
Executing one skill should either:

- Solve the problem.
- Let you execute another skill that could solve the problem.

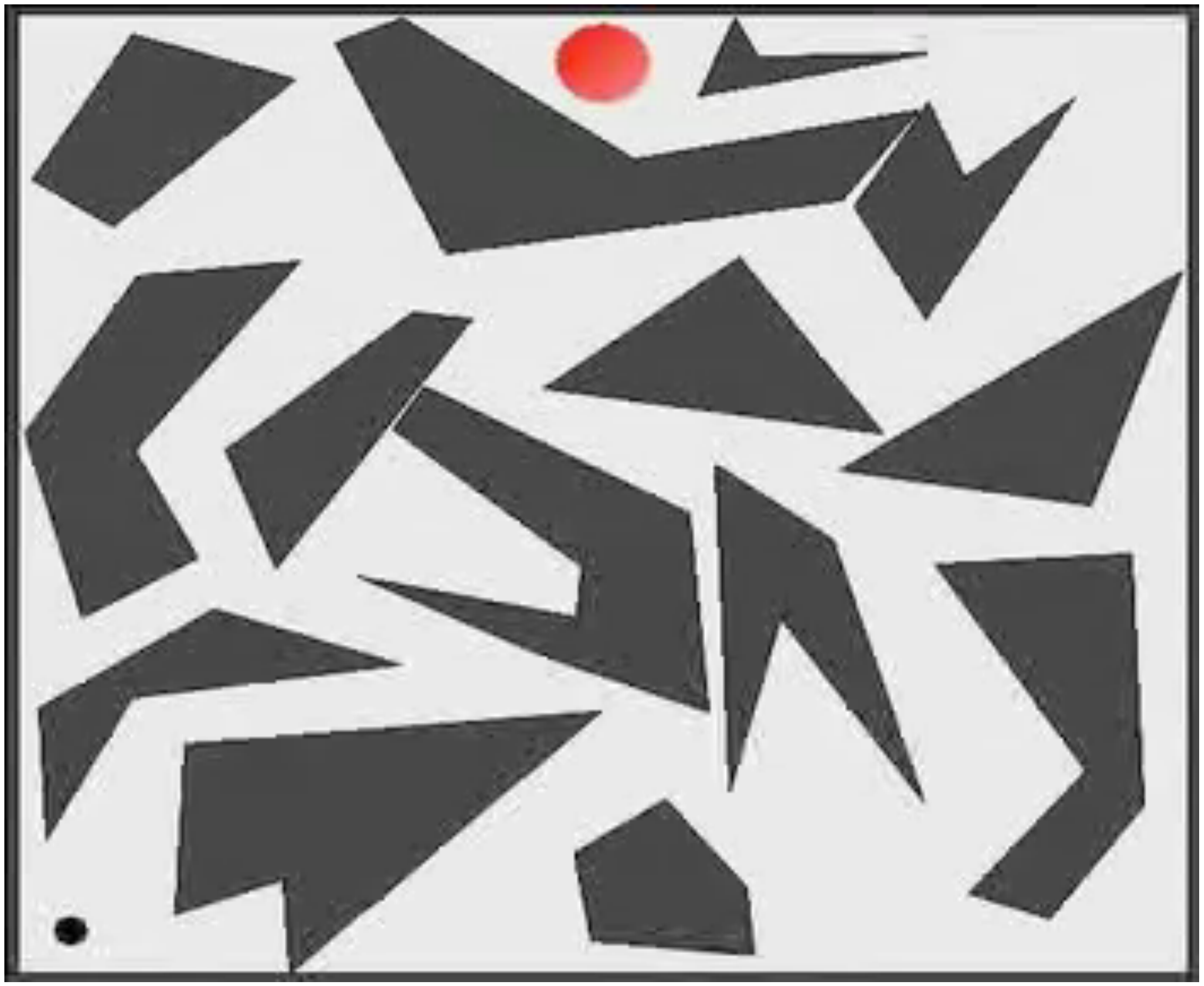
Skills should be *chainable*.

[Konidaris and Barto, NIPS 2009]

Skill Chaining

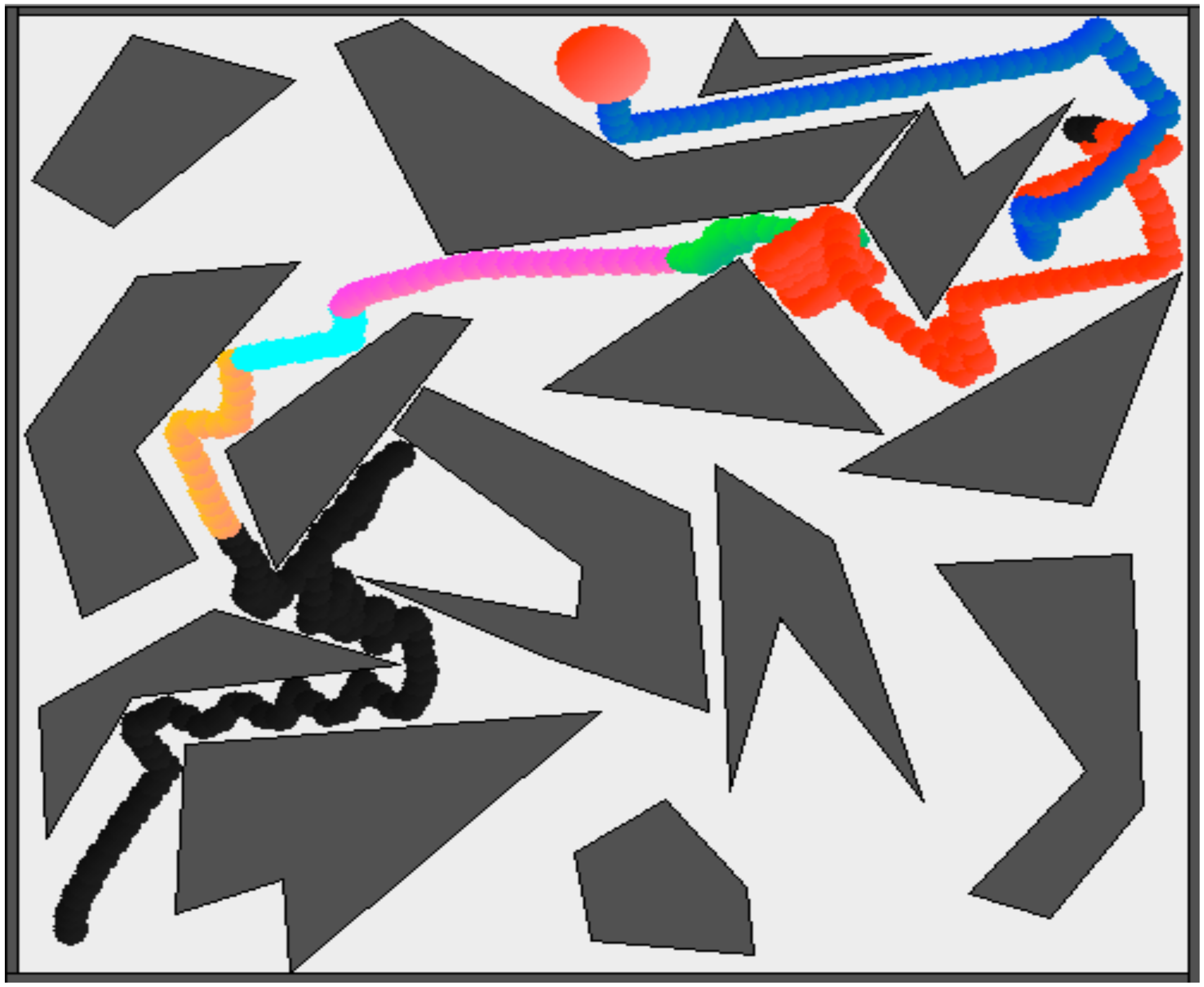


Skill Chaining: Results



[NIPS 2009]

Skill Chaining: Results



[NIPS 2009]

Option-Critic (Bacon et al.)

- Use policy gradient to simultaneously learn high-level policy and option policies / termination conditions
- Assume options can be initiated anywhere
- Options are learned based directly on performance, rather than a heuristic!

Reading Responses

Zayne Sprague
Do we still think Options are needed in RL tasks (I haven't heard of them recently)? Or do we assume (much like with feature extraction) that Neural Networks are learning "Option-like" trajectories by themselves?

Jiaheng Hu
The intra-option learning described in the option-critic paper seems very appealing. However I somehow have the feeling that nowadays people prefer to treat options as fixed policies that are not getting updated during downstream task learning, especially in robotics. Is my feeling right, or am I missing some important literature?

Final Logistics

Next lecture: Abstraction: Options and Hierarchy II
Reading assignments due **2PM Monday**

Final project literature review due at **11:59pm on Thursday, 4/11**