Combining State and Temporal Abstraction

George Konidaris

gdk@cs.duke.edu
Abstraction
Abstraction
Skill Hierarchies

Base control on *skills*.

- Component of behavior.
- Performs continuous, low-level control.
- *Temporal abstraction.*

Some evidence that humans organize their behavior this way.
Skill Hierarchies

Base control on *skills*.

- Component of behavior.
- Performs continuous, low-level control.
- *Temporal abstraction.*

Some evidence that humans organize their behavior this way.
Skill Hierarchies

Base control on *skills*.

- Component of behavior.
- Performs continuous, low-level control.
- *Temporal abstraction*.

Some evidence that humans organize their behavior this way.
Skill Hierarchies

Base control on *skills*.

- Component of behavior.
- Performs continuous, low-level control.
- *Temporal abstraction.*

Some evidence that humans organize their behavior this way.
Skill Hierarchies

*Behavior is modular and compositional.*
Skill Hierarchies

Behavior is modular and compositional.

Skills are like subroutines.

```python
def abs(x):
    if (x > 0):
        return x
    else:
        return -x
```

[Wilkes, Wheeler and Gill, 1951]
Skill-Specific Abstractions

Skills should also be abstract.

- Many high-dimensional problems really are high-dimensional if you try to solve them monolithically.
- Can split into subproblems, each of which support a solution using an abstraction.
Skill-Specific Abstractions

Skills should also be abstract.

- Many high-dimensional problems really are high-dimensional if you try to solve them monolithically.
- Can split into subproblems, each of which support a solution using an abstraction.

[IJCAI 2009]
Skill-Specific Abstractions

Skills should also be abstract.

- Many high-dimensional problems really are high-dimensional if you try to solve them monolithically.
- Can split into subproblems, each of which support a solution using an abstraction.

Behavior is piecewise low-dimensional. [IJCAI 2009]
Skill-Generated Abstractions
Skill-Generated Abstractions
Skills
Skills
Skills
Skills
Skills

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Diagram](image)
The Options Framework

Formal model of a skill.

An option \( o \) is a policy unit:
- Initiation set
- Termination condition
- Option policy

[Sutton, Precup and Singh 1999]
The Options Framework

Formal model of a skill.

An option $o$ is a policy unit:
- Initiation set
- Termination condition
- Option policy

[Sutton, Precup and Singh 1999]
The Options Framework

Formal model of a skill.

An option $o$ is a policy unit:

- Initiation set
- Termination condition
- Option policy

[Sutton, Precup and Singh 1999]
The Options Framework

Formal model of a skill.

An option $o$ is a policy unit:

- Initiation set
- Termination condition
- Option policy

[Sutton, Precup and Singh 1999]
Option Transfer

![Graph showing the number of actions over episodes with different option transfer settings.]

- Perfect Options
- Learned Options
- No Options

[IJCAI 2007]
Skill Chaining
Skill Chaining
Skill Chaining:
Skill Chaining: Results
Key Ideas

What should options do?

Solway et al. [2014] (following Simsek and Barto [2009]):
- Agent faces distribution over future problems.
- Try to maximize performance averaged over distribution.
- Reasonable to use past problems as sample.
Skill Acquisition

- A robot learning to solve a task
- Extracting skills from solution
- Deploying them in a new task
Skill Acquisition

- A robot learning to solve a task
- Extracting skills from solution
- Deploying them in a new task
Training Room

Episode 1 (35x)
Training Room

Episode 1 (35x)
Acquired Skills
Acquired Skills
The Test Room
The Test Room
The Test Room

Median Test Performance Comparison

(50x)  (50x)

Without Acquired Skills    With Acquired Skills
The Test Room

Median Test Performance Comparison

Without Acquired Skills  With Acquired Skills
The Test Room

[AAAI 2011]
Summary

Scaled skill acquisition to real robots:
- Skills extracted because they are useful
- Suitable for further learning (individually)
- Suitable for deployment in new problems

Acquired skills can improve a robot’s problem-solving abilities.
Skill-Generated Representations
Abstraction with Options

Problem difficulty shouldn’t depend on low-level state space.
Abstraction with Options

Problem difficulty shouldn’t depend on low-level state space.
Abstraction with Options

Problem difficulty shouldn’t depend on low-level state space.
Skills Cannot Be The Whole Story

Representation Acquisition:
- How should an agent’s representations change as it acquires new skills?
Skills Cannot Be The Whole Story

Representation Acquisition:

• How should an agent’s representations change as it acquires new skills?

More precisely:

• Assume we have skills (SMDP).
• Can we *automatically derive* an appropriate abstract representation for planning with those skills?
• SMDP to more abstract MDP.
The answer is yes!

We can write down the right abstract representation for planning using any set of skills.

But the representation depends on properties of the skills.
Key Idea

Formalize the fundamental question a representation needs to answer, and then *explicitly construct* it so that it can answer that question.

What is the fundamental question of probabilistic planning?

[AAAI 2014, IJCAI 2015]
Key Idea

Formalize the fundamental question a representation needs to answer, and then *explicitly construct* it so that it can answer that question.

What is the fundamental question of probabilistic planning?

Given a state and a sequence of options \(\{o_1, o_2, \ldots, o_n\}\):

- What is the probability of being able to execute it?
- What is the expected reward?

[AAAI 2014, IJCAI 2015]
Symbols for Planning

A plan $p = \{o_1, ..., o_n\}$ from a state distribution $Z$ is a sequence of actions to be executed from a state drawn from $Z$.

Starting from the corridor ...

- GoToDoor
- TurnHandle
- PushDoorOpen
- EnterRoom ...

So:

- Which distributions do we need to determine the feasibility of any plan $p$?
Symbols for Planning

We need **one distribution** and one operator per skill.

Initiation distribution:

\[ P(s \in I_o) \]
Symbols for Planning

We need one symbol and **one operator** per skill.

Image distribution:

---

**Definition**  Given a start distribution \( Z(S) \) and an option \( o \), we define the probabilistic image of \( o \) from \( Z(S) \) as:

\[
Im(o, Z) = \frac{\int_S P(s'|s, o)Z(s)P(I_o|s) \, ds}{\int_S Z(s)P(I_o|s) \, ds},
\]

where \( P(s'|s, o) = \int P(s', \tau|s, o) \, d\tau \), since we are not concerned with the time taken to execute \( o \).
Probabilistic Planning

Must deal with *distributions over states* in the future.
Probabilistic Planning

Must deal with \textit{distributions over states} in the future.
Probabilistic Planning

Must deal with *distributions over states* in the future.
Subgoal Options

\[ P(s' | o_i, s) = P(s' | o_i) \]
Subgoal Options

Results in a *plan graph*.

- Node for each option.
- Probability of moving from $i$ to $j$
Subgoal Options

Results in a *plan graph*.

- Node for each option.
- Probability of moving from *i* to *j*
Abstract subgoal option:

- \( s = [a, b] \)
- \( a \) (mask) is set to some subgoal distribution.
- \( b \) remains unchanged.

\[
[a, b, c, d, e, f, g, h]
\]

\[
[a, b, c, d, e, f', g', h']
\]
Abstract Subgoal Options

Abstract subgoal option:

- \( s = [a, b] \)
- \( a \) (mask) is set to some subgoal distribution.
- \( b \) remains unchanged.

\[ [a, b, c, d, e, f, g, h] \]

Factored MDP

\[ [a, b, c, d, e, f', g', h'] \]
Abstract MDPs

Abstract subgoal options: can generate factored MDP

- Vocabulary of state factors + forward model
- Provably sound and complete
- Can discard grounding distributions once done
What is a Symbol?

A (propositional) symbol is a name for a set of low-level states.

**Definition**  
A propositional symbol $\sigma_Z$ is the name associated with a test $\tau_Z$, and the corresponding set of states $Z = \{ s \in S \mid \tau_Z(s) = 1 \}$. 
What is a Symbol?

A (propositional) symbol is a name for a set of low-level states.

Definition: A propositional symbol $\sigma_Z$ is the name associated with a test $\tau_Z$, and the corresponding set of states $Z = \{ s \in S \mid \tau_Z(s) = 1 \}$.

$$f(s) = \frac{1}{1 + e^{-\theta \cdot s}}$$
Defining a Symbol

What do operations on our symbols mean?

(Concrete boolean algebra)
Probabilistic Symbols

Learning symbolic representations

- Execute options and get some data
  \((s, o, s', r) (s, I_o?)\)

- For each option:
  - Partition into \sim abstract subgoal options
  - For each partitioned option:
    - Probabilistic classifier for init distribution
    - Density estimator for image distribution
    - Regression for reward model
Probabilistic Symbols
Probabilistic Symbols

- Probability: 0.795
- Probability: 0.205
- Interaction: interact
PPDDL

(:action interact_option
 :parameters ()
 :precondition (and (notfailed) (symbol14)
   (symbol20) (symbol3))
 :effect (probabilistic
   0.7955 (and (symbol21) (symbol22)
     (not (symbol3)))
   0.2045 (and (symbol4))
 )
)

learned PPDDL representation
Symbols

(b) symbol129
(c) symbol128
(d) symbol128 and symbol129

(e) symbol17
(f) symbol120
(g) symbol1
Planning

<table>
<thead>
<tr>
<th>Goal</th>
<th>Min. Depth</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obtain Key</td>
<td>14</td>
<td>35</td>
</tr>
<tr>
<td>Obtain Treasure</td>
<td>26</td>
<td>64</td>
</tr>
<tr>
<td>Treasure &amp; Home</td>
<td>42</td>
<td>181</td>
</tr>
</tbody>
</table>

... using mGPT (Bonet and Geffner, 2005)
Robots
Robots
True Abstraction Hierarchies

Base MDP: \( M_0 = \{ S_0, A_0, R_0, P_0 \} \)

Successive MDPs: \( M_i = \{ S_i, A_i, R_i, P_i \} \)
True Abstraction Hierarchies

Basic assumption of hierarchical RL:

- $A_j$ is a set of options over $M_{j-1}$

\[
M_j = \{S_j, A_j, R_j, P_j\}
\]

options over

\[
M_{j-1} = \{S_{j-1}, A_{j-1}, R_{j-1}, P_{j-1}\}
\]
True Abstraction Hierarchies

Basic assumption of hierarchical RL:

- $A_j$ is a set of options over $M_{j-1}$

\[
M_j = \{S_j, A_j, R_j, P_j\}
\]

\[
M_{j-1} = \{S_{j-1}, A_{j-1}, R_{j-1}, P_{j-1}\}
\]

Now we know what $S_j, R_j, P_j$ must be.
The Skill-Symbol Loop

Skills \(\text{determines}\) Symbols

skill acquisition

Discrete MDP \(\text{describes}\)
The Skill-Symbol Loop

**Skills:**
- Pregrasp
- Grasp
- Lift
- Move Arm to Above Box
- Release

**Factors:**
- Above-Box-Apple
- Pregrasped
- Grasped Apple
- Apple-in-Air
- Arm above B1, B2
- Apple in B1, B2
The Skill-Symbol Loop

New Skills:
- Grab-Apple
- Move Arm to Above Box
- Drop-Apple

Factors:
- Grasped/Lifted Apple
- Arm above B1, B2
- Apple in B1, B2
The Skill-Symbol Loop

New Skills:
MoveAppleTo

Factors:
Apple in B1, B2
Succession of MDPs:

\[ M_i = \{S_i, A_i, R_i, P_i\} \]

As we go up in the hierarchy:

- Symbols more general (refer to broader distributions)
- Eventually reach “basic” problem description.
- Robot details wash out.

*No choice* other than the skill discovery algorithm.
Planning

A solution at any level $i$ is a solution to $M_0$.

Consequently, for a given start and goal set, we need to find highest $i$ (smallest problem) to plan at.
Planning

A solution at any level $i$ is a solution to $M_0$.

Consequently, for a given start and goal set, we need to find highest $i$ (smallest problem) to plan at.
Taxi

Options:
1. up, down, left, right, pick up, drop off
2. drive to each depot, pick up, drop off
3. passenger-to-depot

Hierarchical Planning

<table>
<thead>
<tr>
<th>Query</th>
<th>Level</th>
<th>Matching</th>
<th>Planning</th>
<th>Total</th>
<th>Base + Options</th>
<th>Base MDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>&lt;1</td>
<td>&lt;1</td>
<td>&lt;1</td>
<td>770.42</td>
<td>1423.36</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>&lt;1</td>
<td>10.55</td>
<td>11.1</td>
<td>1010.85</td>
<td>1767.45</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>12.36</td>
<td>1330.38</td>
<td>1342.74</td>
<td>1174.35</td>
<td>1314.94</td>
</tr>
</tbody>
</table>
Summary

Close link between symbolic representation and skills

- Environment + goal + skills specify symbolic representation we need.
- That representation is learnable.

Skills determine the symbols you need to create plans with them.

We can combine skills and high-level representations to achieve true abstraction hierarchies.
Thank you!

Questions?