Planning with Affordances

Abstract

Current methods for exactly solving decision-making under uncertainty require exhaustive enumeration of all possible states and actions, leading to expotential run times which have been referred to as the "curse of dimensionality." Approaches to address this problem by providing the system with formally encoded knowledge such as options or macro-actions, still fail to prevent the system from considering many actions which seem obviously irrelevant for a human partner. To address this issue, we introduce a novel approach to representing knowledge about how to plan in terms of *affordances* [1]. Evaluation is performed in the Minecraft domain on several path planning tasks - we demonstrate a significant increase in speed and reduction in state-space exploration across 5 different path planning tasks.

1 INTRODUCTION

2 BACKGROUND

2.1 AFFORDANCES

An Affordance is defined as: $\mathcal{A} = <$ a, p, $\mathcal{G} >$, where:

$$a \in A$$

$$p \ : s \rightarrow \{0, 1\}$$

$$\mathcal{G} \ : s \rightarrow \{0, 1\}$$

Where a is a *primitive action* in the agent's set of available actions, p is a *precondition* which is a predicate on a given state s, and $\mathcal{G}$ is a desired *postcondition*, which is also a predicate on a particular state s.

The constituents that make up an Affordance parallel those of the other planning approaches discussed in the background section.

2.2 SUBGOALS

Subgoal planning leverages the intuition that certain goals in planning domains may only be brought about if certain preconditions are first satisfied. For instance, in the Minecraft domain, one must be in possession of grain in order to bake bread. In Branavan et. al, they explore learning subgoals and applying them in order to plan through a variety of problems in Minecraft.

In subgoal planning, the agent is given a pair of predicates, $< x_k, x_l >$, where x_l is the effect of some action sequence performed on a state in which x_k is true. Thus, subgoal planning entails high-level planning in subgoal space, and low-level planning to get from subgoal to subgoal.

However, subgoal planning falls short in that sub-goals only go so far in directing the low-level planner. The planner will still waste many cycles searching through meaningless portions of the state-space.

2.3 OPTIONS

The options framework proposes incorporating high-level policies to accomplish specific sub tasks. For instance, when an agent is near a door, the agent can engage the 'door-opening-option-policy', which switches from the standard high-level planner to running a policy that is hand crafted to open doors. An option o is defined as follows:

$o \ = < \pi_0, I_0, \beta_0 >$, where:

$$\pi_0 : (s, a) \rightarrow [0, 1]$$

$$I_0 : s \rightarrow \{0, 1\}$$

$$\beta_0 : s \rightarrow [0, 1]$$

Here, π_0 represents the *option policy*, I_0 represents a precondition, under which the option policy may initiate, and β_0 represent the post condition, which determines which states terminate the execution of the option policy.

As Konidaris and Barto point out, the classic options framework is not generalizable, as it does not enable an agent to transfer knowledge from one state space to another. Recently, Konidaris and Barto's expand on the classic options framework and allow for a more portable implementation of options. Still, though, planning with options requires either that we plan in a mixed space of actions *and* options (which blows up the size of the search space), or requires that we plan entirely in the space of options. Additionally, providing an agent with an option policy is a difficult task for a human designer (especially if we want an optimal policy, which we do).

The Affordance formalism introduced above and expanded on in this paper resolves the weaknesses of these other frameworks by limiting the complexity of the seed knowledge required of the designer, providing enough knowledge to limit the search space, and still maintains scalability. (note: assuming experiments back this up)

3 EXPERIMENTS

Task List, apply each planning system (Aff, O, SG) to all tasks.

4 RESULTS

	Affordances	Options	Subgoals
Task One	1	1	0
Task Two	1	1	1
Task Three	1	1	0

5 CONCLUSION

References

[1] JJ Gibson. The concept of affordances. *Perceiving, acting, and knowing*, pages 67–82, 1977.