Skill Chaining: Skill Discovery in Continuous Domains

George Konidaris GDK@CS.UMASS.EDU
Andrew Barto BARTO@CS.UMASS.EDU
Autonomous Learning Laboratory, Computer Science Dept., University of Massachusetts Amherst, 01003 USA

1. Introduction

An important research goal for hierarchical reinforcement learning is the development of methods by which an agent can discover new skills autonomously, and thereby build its own high-level skill hierarchies. Although several methods exist for skill discovery in discrete domains, none are immediately extensible to or have been successfully applied in continuous domains.

We introduce skill chaining, a skill discovery method for continuous domains. Skill chaining produces chains of skills leading to a salient event—where salience can be defined simply as an end-of-task reward, or as a more sophisticated heuristic (e.g., an intrinsically interesting event (Singh et al., 2004)). The goal of each skill in the chain is to reach a state where its successor skill can be executed.

2. Skill Discovery in Continuous Domains

Two difficulties that are present but less apparent in discrete domains become more immediate in continuous domains. First, all existing skill discovery methods identify a single goal state as an option target, whereas in continuous domains this must be generalized to a *target region*. Second, while in discrete domains an option's initiation set may expand arbitrarily as the agent learns the option policy, in continuous domains the policy must always be approximated and thus may be only locally applicable.

Since a useful option lies on the solution path, it seems intuitively obvious that the first option we create should have the aim of consistently reaching the goal. We can then learn (using a supervised learning method) its initiation set using states from which it executes successfully as positive examples and those from which it does not as negative examples. The high likelihood that its initiation set will be a local neighbourhood around the goal suggests a follow-on step: we could create an option whose goal is to *reach a state where the first option can be executed.*

We term this method *skill chaining*, because repeating this procedure results in a chain of skills leading from the start state to the termination region, as depicted in Figure 1.

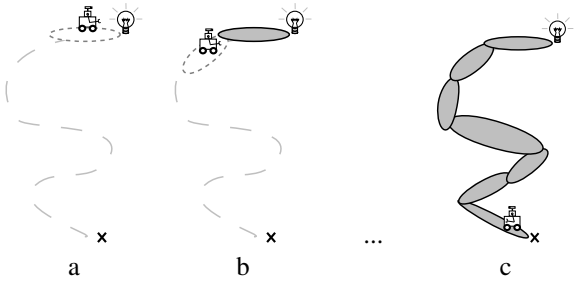

Figure 1. An agent creates skills using skill chaining. First, the agent encounters the goal and creates a skill to reach it (a). Entering the first skill then becomes a new subgoal, which later triggers the creation of a second skill to reach the first (b). Finally, after many trajectories the agent has created a chain of skills to reach the goal (c).

This method can be easily generalized to multiple trajectories (skill trees, rather than chains), and to use more general target events than the goal.

3. Empirical Evaluation

Our experiments use the Pinball domain,[1] where an agent must maneuver a ball into a whole through a domain with several obstacles. The ball is dynamic, so its state is described by four variables: x, y, $\dot{x}$ and $\dot{y}$, and collisions are fully elastic. The agent may add or subtract a small force to $\dot{x}$ or $\dot{y}$ (which incurs a reward of -5 per action), or leaving them unchanged (which incurs a reward of -1 per timestep); reaching the goal obtains a reward of $10,000$.

The Pinball domain is an appropriate continuous domain for skill discovery because its sharp discontinuities and extended dynamic control characteristics make it difficult for control and for function approximation—much more difficult than a simple navigation task.

To learn to solve the overall task, we used Sarsa ($\alpha = 0.001, \gamma = 1, \epsilon = 0.01$) (Sutton & Barto, 1998) with a 4th-

[1] Java source code for the PinBall domain is available at http://www-all.cs.umass.edu/~gdk/pinball/

order Fourier Basis (Konidaris & Osentoski, 2008). Each learned option used Q-learning ($\alpha = 0.001, \gamma = 1, \epsilon = 0.01$) with a 3rd-order Fourier Basis. In both cases we varied α systematically to maximize performance. Initiation sets were learned by logistic regression.

Figure 2 shows the performance (averaged over 100 runs) in the Pinball domain for agents using a random policy and flat agents (who do not use any options) against agents employing skill chaining, and agents starting with pre-learned options. The agents with pre-learned options acquired them using skill chaining over 250 episodes of solving the Pinball task, whereupon their task value functions were reset and no new options could be acquired. The agents employing skill chaining perform significantly better than flat agents, and that skills learned via skill chaining significantly improve initial and final performance when given to "blank slate" agents, demonstrating that it is the skills themselves, rather than some byproduct of them, that is responsible for the performance improvement. Figure 3 shows a sample solution trajectory for our PinBall domain, with the options executed shown in different colors.

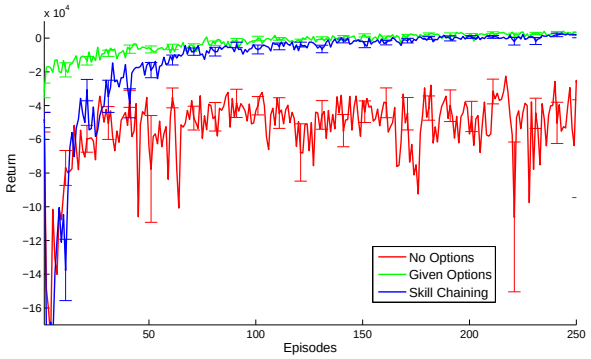

Figure 2. Performance in the Pinball domain (averaged over 100 runs) for agents with a random policy, agents employing skill chaining, agents with given options, and agents without options.

4. Summary and Discussion

The performance gains demonstrated in the previous section show that skill chaining (at least using end-of-episode salience) can significantly improve the performance of a reinforcement learning agent in a challenging continuous domain, by breaking the solution into subtasks and learning lower-order option policies for each one.

An avenue that may lead to further performance benefits would be to include a more sophisticated notion of salience: *any* indicator function can be substituted for (or added to) the end-of-episode one used in this paper. Skill chaining can then be used augment the options discovered using such a method, which we expect will still be low-range.

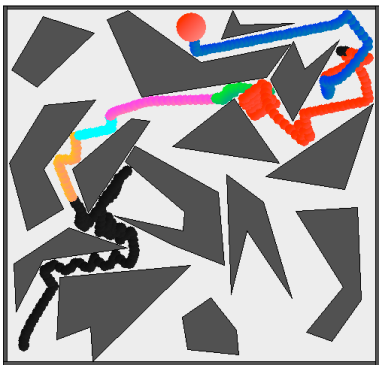

Figure 3. A good solution to the Pinball domain, showing the acquired skills executed along the sample trajectory in different colors. Primitive actions are shown in black.

The primary benefit of skill chaining is that it reduces the representational burden of the overall value function and achieve a better overall solution by allowing each option to represent its own local policy. This implies that skill acquisition is best suited to high-dimensional problems where a single policy cannot be well represented using a feasible number of basis functions in feasible time. In very complex domains such as robotics, where the state space may contain hundreds or even thousands of state variables, we may require a more sophisticated approach that exploits the idea that although the entire task may not be reducible to a feasibly sized state space, it is often possible to split the problem into subtasks that are. One such approach is *abstraction selection* (Konidaris & Barto, 2009), where the agent has a set of possible abstractions (e.g., sets of related features) from which it can select when creating a new option, based on an initial sample trajectory (as is obtained the first time an agent reaches a salient event). We conjecture that the ability of each option to use its own abstraction will be a key benefit to hierarchical reinforcement learning as we try to scale up to high-dimensional problems.

References

Konidaris, G., & Barto, A. (2009). Efficient skill learning using abstraction selection. *Proceedings of the Twenty First International Joint Conference on Artificial Intelligence*.

Konidaris, G., & Osentoski, S. (2008). *Value function approximation in reinforcement learning using the Fourier basis* (Technical Report UM-CS-2008-19). Department of Computer Science, University of Massachusetts, Amherst.

Singh, S., Barto, A., & Chentanez, N. (2004). Intrinsically motivated reinforcement learning. *Proceedings of the 18th Annual Conference on Neural Information Processing Systems*.

Sutton, R., & Barto, A. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.