Intra-Option Learning about Temporally Abstract Actions

Richard S. Sutton, Doina Precup
Department of Computer Science
University of Massachusetts
Amherst, MA 01003-4610

Satinder Singh
Department of Computer Science
University of Colorado
Boulder, CO 80309-0430

Abstract

Several researchers have proposed modeling temporally abstract actions in reinforcement learning by the combination of a policy and a termination condition, which we refer to as an ”option”. Value functions over options and models of options can be learned using methods designed for semi-Markov decision processes (SMDPs). However, these methods all require an option to be executed to termination. In this paper we explore methods that learn about an option from small fragments of experience consistent with that option, even if the option itself is not executed. We call these methods ”intra-option” learning methods because they learn from experience within an option. Intra-option methods are sometimes much more efficient than SMDP methods because they can use off-policy temporal-difference mechanisms to learn simultaneously about all the options consistent with an experience, not just the few that were actually executed. In this paper we present intra-option learning methods for learning value functions over options and for learning multi-step models of the consequences of options. We present computational examples in which these new methods learn much faster than SMDP methods and learn effectively when SMDP methods cannot learn at all. We also sketch a convergence proof for intra-option value learning.

Keywords: Reinforcement Learning, Temporal Abstraction, Hierarchical Learning Semi-Markov Decision Processes, Model Learning
E-mail: rich@cs.umass.edu
Phone: 978-897-6174
Multiple submission statement: This paper contains results that are also being prepared to form part of a larger paper to be submitted for journal publication.

1 Introduction

The incorporation of temporally extended courses of actions into reinforcement learning algorithms has been one of the focuses of recent research in the field (e.g. Dayan, 1993;; Dayan and Hinton, 1992;; Dietterich, 1997, Huber and Grupen, 1997;; Kaelbling, 1993; Mahadevan and Connell, 1992; McGovern, Sutton and Fagg, 1997; Parr and Russel, 1998; Precup & Sutton, 1998; Precup, Sutton & Singh, 1998; Singh, 1992Sutton, 1995). A temporally extended course of action is defined by a set of states in which it applies, an internal policy that it executes and a termination condition. We will use the term ”option” to denote such a course of action.

Options define a more abstract temporal level for planning and learning. SMDP learning and planning methods can be applied at this higher level, in order to obtain significant speed improvements over one-step methods. However, SMDP methods treat temporally extended course of action as opaque black boxes. Therefore, a learning agent has to actually execute each option to completion in order to gather information about its outcomes. Because of this property, SMDP methods can only be used for terminating courses of action, and can only learn about one such option at one time.

In this paper, we present learning algorithms that circumvent these inconveniences, by taking advantage of the observation that the SMDP generated by the options is based in an inderlying MDP. Therefore, we can learn about the effects of temporally extended actions by relying on temporal difference methods, without requiring that an option is executed to completion. We such learning methods *intra-option* methods because they learn from experience within a single option. Intra-option methods can even be used to learn about the model of an option without ever executing the option, as long as some selections are made that are consistent with it. Intra-option methods are examples of *off-policy* learning methods (Sutton and Barto, 1998) because they learn about the consequences of one policy while actually behaving according to another, potentially different policy. They can be used simultaneously to learn about several different options at the same time.

We present intra-option learning algorithms both for learning the values of the options and for learning their models. We give a proof sketch for the convergence of intra-option learning. The flexibility and speed gains of these methods are illustrated through small computational experiments.

2 Reinforcement Learning (MDP) Framework

First we briefly summarize the mathematical framework of the reinforcement learning problem that we use in the paper. In this framework, a learning *agent* interacts with an *environment* at some discrete, lowest-level time scale $t = 0, 1, 2, \ldots$. At each time step, the agent perceives the state of the environment, s_t, and on that basis chooses a primitive action, a_t. In response to each primitive action, a_t, the environment produces one step later a numerical reward, r_{t+1}, and a next state, s_{t+1}. We denote by $\mathcal{A} = \bigcup_{s \in \mathcal{S}} \mathcal{A}_s$ the union of the action sets. If $\mathcal{S}$ and $\mathcal{A}$, are finite, then the environment's transition dynamics are modeled by one-step state-transition probabilities, and one-step expected rewards,

$$p^a_{ss'} = \Pr\{s_{t+1} = s' \mid s_t = s, a_t = a\} \quad \text{and} \quad r^a_s = E\{r_{t+1} \mid s_t = s, a_t = a\}, \quad (1)$$

for all $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$ (it is understood here that $p^a_{ss'} = 0$ for $a \notin \mathcal{A}_s$). These two sets of quantities together constitute the *one-step model* of the environment.

The agent's objective is to learn a policy π, which is a mapping from states to probabilities of taking each action, that maximizes the expected discounted future reward from each state s:

$$V^\pi(s) = E\Big\{r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots \Bigm| s_0 = s, \pi\Big\},$$

where $\gamma \in [0, 1)$ is a *discount-rate* parameter. The quantity $V^\pi(s)$ is called the *value* of state s under policy π, and V^π is called the value function for policy π. The optimal value of a state is denoted

$$V^*(s) = \max_\pi V^\pi(s)$$

Particularly important for learning methods is a parallel set of value functions for state–action pairs rather than for states. The value of taking action a in state s under policy π, denoted $Q^\pi(s, a)$, is the expected discounted future reward starting in s, taking a, and henceforth following π:

$$Q^\pi(s, a) = E\Big\{r_{t+1} + \gamma r_{t+1} + \gamma^2 r_{t+1} + \cdots \Bigm| s_t = s, a_t = a, \pi\Big\} \quad (2)$$

This is known as the *action-value function* for policy π. The *optimal* action-value function is

$$Q^*(s, a) = \max_\pi Q^\pi(s, a)$$

The action value functions satisfy the Bellman equations:

$$Q^\pi(s,a) = r_s^a + \gamma \sum_{s'} p_{ss'}^a \sum_{a'} \pi(s,a) Q^\pi(s',a') \tag{3}$$

$$Q^*(s,a) = r_s^a + \gamma \sum_{s'} p_{ss'}^a \max_{a'} Q^*(s',a') \tag{4}$$

3 Options

We use the term *options* for the generalization of primitive actions to include temporally extended courses of action. In this paper, we focus on *Markov options* which consist of three components: a policy $\pi : \mathcal{S} \times \mathcal{A} \mapsto [0,1]$, a termination condition $\beta : \mathcal{S} \mapsto [0,1]$, and an input set $\mathcal{I} \subseteq \mathcal{S}$. An option $< \pi, \beta, \mathcal{I} >$ is available in state s if and only if $s \in \mathcal{I}$. If the option is taken, then actions are selected according to π until the option terminates stochastically according to β. In particular, if the option taken in state s_t is Markov, then the next action a_t is selected according to the probability distribution $\pi(s,\cdot)$. The environment then makes a transition to state s_{t+1}, where the option either terminates, with probability $\beta(s_{t+1})$, or else continues, determining a_{t+1} according to $\pi(s_{t+1},\cdot)$, possibly terminating in s_{t+2} according to $\beta(s_{t+2})$, and so on. When the option terminates, then the agent has the opportunity to select another option.

The input set and termination condition of an option together restrict its range of application in a potentially useful way. In particular, they limit the range over which the option's policy need be defined. For example, a handcrafted policy π for a mobile robot to dock with its battery charger might be defined only for states $\mathcal{I}$ in which the battery charger is within sight. The termination condition β would be defined to be 1 outside of $\mathcal{I}$ and when the robot is successfully docked. For Markov options it is natural to assume that all states where an option might continue are also states where the option might be taken (i.e., that $\{s : \beta(s) < 1\} \subseteq \mathcal{I}$). In this case, π need only be defined over $\mathcal{I}$ rather than over all of $\mathcal{S}$.

Given a set of options, their input sets implicitly define a set of available options $\mathcal{O}_s$ for each state $s \in \mathcal{S}$. $\mathcal{O}_s$ are much like the sets of available actions, $\mathcal{A}_s$. We can unify these two kinds of sets by noting that actions can be considered a special case of options. Each action a corresponds to an option that is available whenever a is available ($\mathcal{I} = \{s : a \in \mathcal{A}_s\}$), that always lasts exactly one step ($\beta(s) = 1,\ \forall s \in \mathcal{S}$), and that selects a everywhere ($\pi(s,a) = 1,\ \forall s \in \mathcal{I}$). Thus, we can consider the agent's choice at each time to be entirely among options, some of which persist for a single time step, others which are more temporally extended. The former we refer to as *one-step* or *primitive* options and the latter as *multi-step* options.

Because options terminate in a well defined way, we can consider sequences of them in much the same way as we consider sequences of actions. We can consider policies that select options instead of primitive actions, and we can model the consequences of selecting an option much as we model the results of an action. When initiated in a state s_t, the Markov policy over options $\mu : \mathcal{S} \times \mathcal{O} \mapsto [0,1]$ selects an option $o \in \mathcal{O}_s$ according to probability distribution $\mu(s_t, \cdot)$. The option o is then taken in s_t, determining actions until it terminates in s_{t+k}, at which point a new option is selected, according to $\mu(s_{t+k}, \cdot)$, and so on. In this way a policy over options, μ, determines a conventional policy over actions, or *flat policy*, $\pi = f(\mu)$. Henceforth we use the unqualified term *policy* for policies over options, which include flat policies as a special case.

The definitions of state and action values can be generalized to apply to general policies and options. The value of a state s under a flat policy π is the expected return if the policy is started in s:

$$V^\pi(s) \stackrel{\text{def}}{=} E\Big\{r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots \;\Big|\; \mathcal{E}(\pi, s, t)\Big\}, \tag{5}$$

where $\mathcal{E}(\pi, s, t)$ denotes the event of π being initiated in s at time t. The value of a state under a general policy μ can then be defined as the value of the state under the corresponding flat policy: $V^\mu(s) \stackrel{\text{def}}{=} V^{f(\mu)}$.

It is natural to the generalize action-value function to an *option*-value function. We define $Q^\mu(s, o)$, the value of taking option o in state $s \in \mathcal{I}$ under policy μ, as

$$Q^\mu(s, o) \stackrel{\text{def}}{=} E\Big\{r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots \;\Big|\; \mathcal{E}(o\mu, s, t)\Big\}, \tag{6}$$

where $o\mu$, the *composition* of o and μ, denotes the policy that first follows o until it terminates and then initiates μ in the resultant state.

4 SMDP (Option-to-Option) Learning

Options are closely related to the actions in a special kind of decision problem known as a *semi-Markov decision process*, or *SMDP* (e.g., see Puterman, 1994). In fact, any MDP with a fixed set of options *is* an SMDP.

Planning with options of course requires a model of their effects. Fortunately, the appropriate form of model for options, analogous to the r_s^a and $p_{ss'}^a$ defined earlier for actions, is known from existing SMDP theory. For each state in which an option may be started, this kind of model predicts the state in which the option will terminate and the total reward received along the way. These quantities are discounted in a particular way. For any option o, let $\mathcal{E}(o,s,t)$ denote the event of o being taken in state s at time t. Then the reward part of the model of o for state s is

$$r_s^o = E\Big\{r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{k-1} r_{t+k} \;\Big|\; \mathcal{E}(o,s,t)\Big\}, \tag{7}$$

where $t+k$ is the random time at which o terminates. The state-prediction part of the model of o for state s is

$$p_{ss'}^o = \sum_{j=0}^{\infty} \gamma^j \Pr\{k=j, s_{t+j}=s' \mid \mathcal{E}(o,s,t)\} = E\Big\{\gamma^k \delta_{s' s_{t+k}} \mid \mathcal{E}(o,s,t)\Big\}, \tag{8}$$

for all $s' \in \mathcal{S}$, under the same conditions, where $\delta_{ss'}$ is an identity indicator, equal to 1 if $s = s'$, and equal to 0 else. Thus, $p_{ss'}^o$ is a combination of the likelihood that s' is the state in which o terminates together with a measure of how delayed that outcome is relative to γ. We call this kind of model a *multi-step model* because it describes the outcome of an option not at a single time but at potentially many different times, appropriately combined. Using these models we can write Bellman equations for general policies and options. For the purpose of this paper, we focus on the Bellman equation for the value of an option o in state $s \in \mathcal{I}$ under a Markov policy μ:

$$Q^\mu(s,o) = r_s^o + \sum_{s'} p_{ss'}^o \sum_{o' \in \mathcal{O}_s} \mu(s',o') Q^\mu(s',o'). \tag{9}$$

This equation specializes to (3) in the special case in which μ is a conventional policy and o is a conventional action.

The *optimal* value functions and *optimal* Bellman equations can also be generalized to options and to policies over options. Of course, the conventional optimal value functions V^* and Q^* are not affected by the introduction of options; one can ultimately do just as well with primitive actions as one can with options. Nevertheless, it is interesting to know how well one can do with a restricted set of options that does not include all the actions. For example, one might first consider only high-level options in order to find an approximate solution quickly. Let us denote the restricted set of options by $\mathcal{O}$ and the set of all policies selecting only from options in $\mathcal{O}$ by $\Pi(\mathcal{O})$. Then the optimal value function given that we can select only from $\mathcal{O}$ is

$$V^*_{\mathcal{O}}(s) \stackrel{\text{def}}{=} \max_{\mu \in \Pi(\mathcal{O})} V^\mu(s) = \max_{o \in \mathcal{O}_s} E\Big\{r + \gamma^k V^*_{\mathcal{O}}(s') \;\Big|\; \mathcal{E}(o,s)\Big\} \tag{10}$$

where $\mathcal{E}(o,s)$ denotes the event of starting the execution of option o in state s. The optimal option values are defined as:

$$Q^*_{\mathcal{O}}(s,o) \stackrel{\text{def}}{=} \max_{\mu \in \Pi(\mathcal{O})} Q^\mu(s,o) = E\Big\{r + \gamma^k \max_{o' \in \mathcal{O}_{s_{t+k}}} Q^*_{\mathcal{O}}(s',o') \;\Big|\; \mathcal{E}(o,s)\Big\}.$$

Given a set of options, $\mathcal{O}$, a corresponding *optimal policy*, denoted $\mu^*_{\mathcal{O}}$, is any policy that achieves $V^*_{\mathcal{O}}$, i.e., for which $V^{\mu^*_{\mathcal{O}}}(s) = V^*_{\mathcal{O}}(s)$ in all states $s \in \mathcal{S}$. If $V^*_{\mathcal{O}}$ and models of the options are known, then optimal policies can be formed by choosing in any proportion among the maximizing options in (10). Or, if $Q^*_{\mathcal{O}}$ is known, then optimal policies can be formed by choosing in each state s in any proportion among the options o for which $Q^*_{\mathcal{O}}(s,o) = \max'_o Q^*_{\mathcal{O}}(s,o')$. Thus, computing approximations to $V^*_{\mathcal{O}}$ or $Q^*_{\mathcal{O}}$ become the primary goals of planning and learning methods with options.

The problem of finding an optimal policy over a set of options $\mathcal{O}$ can be addressed by learning methods. Because the MDP augmented by the options is an SMDP, we can apply SMDP learning methods as developed by Bradtke and Duff (1995), Parr and Russell (1998; Parr, 1998), Mahadevan (1997), and McGovern, Sutton and Fagg (1997). Each option is viewed as an indivisible, opaque unit. When the execution of option o is started in state s, we next jump to the state s' in which o terminates. Based on this experience, an option-value function $Q(s,o)$ is updated. For example, the SMDP version of one-step Q-learning (Bradtke and Duff, 1995), which we call *SMDP Q-learning*, updates after each option termination by

$$Q(s,o) \leftarrow Q(s,o) + \alpha \left[r + \gamma^k \max_{a \in \mathcal{O}} Q(s',a) - Q(s,o)\right], \tag{11}$$

where k denotes the number of time steps elapsing between s and s', r denotes the cumulative discounted reward over this time, and it is implicit that the step-size parameter α may depend arbitrarily on the states, option, and time steps. The estimate $Q(s,o)$ converges to $Q^*_{\mathcal{O}}(s,o)$ for all $s \in \mathcal{S}$ and $o \in \mathcal{O}$ under conditions similar to those for conventional Q-learning (Parr, 1998).

5 Intra-Option Value Learning

One drawback to SMDP learning methods is that they need to execute an option to termination before they can learn about it. Because of this, they can only be applied to one option at a time–the option that is executing at that time. More interesting and potentially more powerful methods are possible by taking advantage of the structure inside each option. In particular, if the options are Markov and we are willing to look *inside* them, then we can use special temporal-difference methods to learn usefully about an option before the option terminates. This is the main idea behind *intra-option* methods.

Intra-option methods are examples of *off-policy* learning methods (Sutton and Barto, 1998) because they learn about the consequences of one policy while actually behaving according to another, potentially different policy. Intra-option methods can be used to simultaneously learn about many different options from the same experience. Moreover, they can learn about the values of executing certain options *without ever executing* those options.

Intra-option methods for value learning are potentially more efficient than SMDP methods because they extract more training examples from the same experience. For example, suppose we are learning to approximate $Q^*_{\mathcal{O}}(s, o)$ and that o is Markov. Based on an execution of o from t to $t + k$, SMDP methods extract a single training example for $Q^*_{\mathcal{O}}(s, o)$. But because o is Markov, it is, in a sense, also initiated at each of the steps between t and $t + k$. The jumps from each intermediate s_i to s_{t+k} are also valid experiences with o, experiences that can be used to improve estimates of $Q^*_{\mathcal{O}}(s_i, o)$. Or consider an option that is very similar to o and which would have selected the same actions, but which would have terminated one step later, at $t + k + 1$ rather than at $t + k$. Formally this is a different option, and formally it *was not executed*, yet all this experience could be used for learning relevant to it. In fact, an option can often learn something from experience that is only slightly related (occasionally selecting the same actions) to what would be generated by executing the option. This is the idea of off-policy training—to make full use of whatever experience occurs in order to learn as much possible about all options, irrespective of their role in generating the experience. To make the best use of experience we would like an off-policy and intra-option version of Q-learning.

It is convenient to introduce new notation for the value of a state–option pair given that the option is Markov and executing upon *arrival* in the state:

$$\tilde{Q}^*_{\mathcal{O}}(s,o) = (1-\beta(s))Q^*_{\mathcal{O}}(s,o) + \beta(s)\max_{o'\in\mathcal{O}} Q^*_{\mathcal{O}}(s,o'), \tag{12}$$

Then we can write Bellman-like equations that relate $Q^*_{\mathcal{O}}(s,o)$ to expected values of $\tilde{Q}^*_{\mathcal{O}}(s',o)$, where s' is the immediate successor to s after initiating Markov option $o =< \pi, \beta, \mathcal{I} >$ in s:

$$\begin{aligned} Q^*_{\mathcal{O}}(s,o) &= \sum_{a\in\mathcal{A}_s} \pi(s,a) E\Big\{r + \gamma \tilde{Q}^*_{\mathcal{O}}(s',o) \;\Big|\; s,a\Big\} \\ &= \sum_{a\in\mathcal{A}_s} \pi(s,a)\left[r^a_s + \sum_{s'} p^a_{ss'} \tilde{Q}^*_{\mathcal{O}}(s',o)\right], \end{aligned} \tag{13}$$

where r is the immediate reward upon arrival in s'. Now consider learning methods based on this Bellman equation. Suppose action a_t is taken in state s_t to produce next state s_{t+1} and reward r_{t+1}, and that a_t was selected in a way consistent with the Markov policy π of an option $o =< \pi, \beta, \mathcal{I} >$. That is, suppose that a_t was selected according to the distribution $\pi(s_t,\cdot)$. Then the Bellman equation above suggests applying the off-policy one-step temporal-difference update:

$$Q(s_t,o) \leftarrow Q(s_t,o) + \alpha\Big[(r_{t+1} + \gamma \tilde{Q}^*_{\mathcal{O}}(s_{t+1},o)) - Q(s_t,o)\Big], \tag{14}$$

The method we call *one-step intra-option Q-learning* applies this update rule to every option o consistent with every action taken a_t.

Theorem 1 (Convergence of intra-option Q-learning) *For any set of deterministic Markov options $\mathcal{O}$, one-step intra-option Q-learning converges w.p.1 to the optimal Q-values, $Q^*_{\mathcal{O}}$, for every option regardless of what options are executed during learning provided every primitive action gets executed in every state infinitely often.*

Proof: (Sketch) On experiencing $< s,a,r,s' >$, for every option o that picks action a in state s, intra-option Q-learning performs the following update:

$$Q_{\mathcal{O}}(s,o) \leftarrow Q_{\mathcal{O}}(s,o) + \alpha(s,o)[r + \gamma\tilde{Q}_{\mathcal{O}}(s',o) - Q_{\mathcal{O}}(s,o)], \quad \text{where}$$

$$\tilde{Q}_{\mathcal{O}}(s',o) = (1-\beta(s'))Q_{\mathcal{O}}(s',o) + \beta(s')\max_{o'\in O} Q_{\mathcal{O}}(s',o').$$

Let $\pi(s)$ be the action selection by deterministic Markov option $o =< \pi, \beta, \mathcal{I} >$. Our result follows directly from Theorem 1 of Jaakkola et al. (1994) and the observation that the expected value of the update operator $r + \gamma \tilde{Q}_{\mathcal{O}}(s', o)$ yields a contraction, as shown below:

$$
\begin{aligned}
|E\{r + \gamma \tilde{Q}_{\mathcal{O}}(s', o)\} - Q^*_{\mathcal{O}}(s, o)| \;&=\; |r_s^{\pi(s)} + \sum_{s'} p_{ss'}^{\pi(s)} \tilde{Q}_{\mathcal{O}}(s', o) - Q^*_{\mathcal{O}}(s, o)| \\
&= |r_s^{\pi(s)} + \sum_{s'} p_{ss'}^{\pi(s)} \tilde{Q}_{\mathcal{O}}(s', o) - r_s^{\pi(s)} - \sum_{s'} p_{ss'}^{\pi(s)} \tilde{Q}^*_{\mathcal{O}}(s', o)| \\
&\leq |\sum_{s'} p_{ss'}^{\pi(s)} \Big[(1 - \beta(s'))(Q_{\mathcal{O}}(s', o) - Q^*_{\mathcal{O}}(s', o)) \\
&\qquad + \beta(s')(\max_{o' \in \mathcal{O}} Q_{\mathcal{O}}(s', o') - \max_{o' \in \mathcal{O}} Q^*_{\mathcal{O}}(s', o'))\Big]| \\
&\leq \sum_{s'} p_{ss'}^{\pi(s)} \max_{s'', o''} |Q_{\mathcal{O}}(s'', o'') - Q^*_{\mathcal{O}}(s'', o'')| \\
&\leq \gamma \max_{s'', o''} |Q_{\mathcal{O}}(s'', o'') - Q^*_{\mathcal{O}}(s'', o'')| \qquad \diamond
\end{aligned}
$$

6 Illustration

As an illustration of the intra-option value learning method, we will use the grid-world environment shown in Figure 1. The cells of the grid correspond to the states of the environment. From any state the agent can perform one of four actions, `up`, `down`, `left` or `right`, which have a stochastic effect. With probability 2/3, the actions cause the agent to move one cell in the corresponding direction, and with probability 1/3, the agent moves instead in one of the other three directions, each with 1/9 probability. If the movement would take the agent into a wall, then the agent remains in the same cell. There are small negative rewards for each action, with means uniformly distributed between 0 and -1. The rewards are also perturbed by gaussian noise with standard deviation 0.1. The environment also has a goal state, labeled “G”. Whenever the agent enters “G”, it gets a reward of 1 and transitions from into a terminal state. The discount parameter was $\gamma = 0.9$.

In each of the four rooms we provide two built-in *hallway options* designed to take the agent from anywhere within the room to one of the two hallway cells leading out of the room. The policies underlying the options allow the agent to take the shortest path to the hallway.

For the first experiment, we applied the intra-option method in this environment,

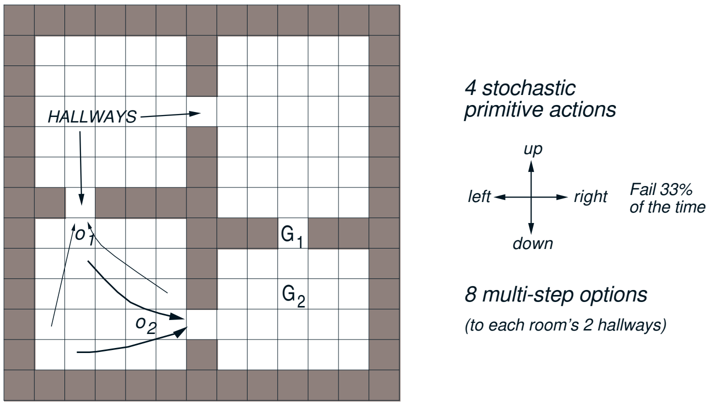

Figure 1: The rooms example is a gridworld environment with stochastic cell-to-cell actions and room-to-room hallway options. Two of the hallway options are suggested by the arrows labeled o_1 and o_2. The label G indicates the location used as goal in experiments described in this section.

without ever selecting the hallway options. On each episode, the agent starts at a random state in the environment and selects primitive actions randomly, with equal probability. This is a case in which SMDP methods would not be able to learn anything about the hallway options, since these options are never executed. However, intra-option learning can learn the values of these actions successfully, as shown in Figure 2. The left panel shows the absolute error between the learned and true values of $Q^*_{\mathcal{O}}$, averaged over $\mathcal{I}$, and over 30 repetitions of the whole experiment, for each of the eight hallway options. The right panel shows the correct and learned values for the two hallway options that apply in the state marked as S in Figure 1. Similar convergence to the true values can be observed for any of the states and options in the environment.

So far we have illustrated the success of intra-option learning in a task in which SMDP methods do not apply. But how do intra-option methods compare to SMDP methods when both are applicable? In order to investigate this question, we used the same environment, but now we allowed the agent to choose among the primitives, as well as the hallway options. In this case, SMDP methods can be applied, since all the options are actually executed. We experimented with two SMDP methods: SMDP Q-learning (Bradtke and Duff, 1995) and a form of hierarchical

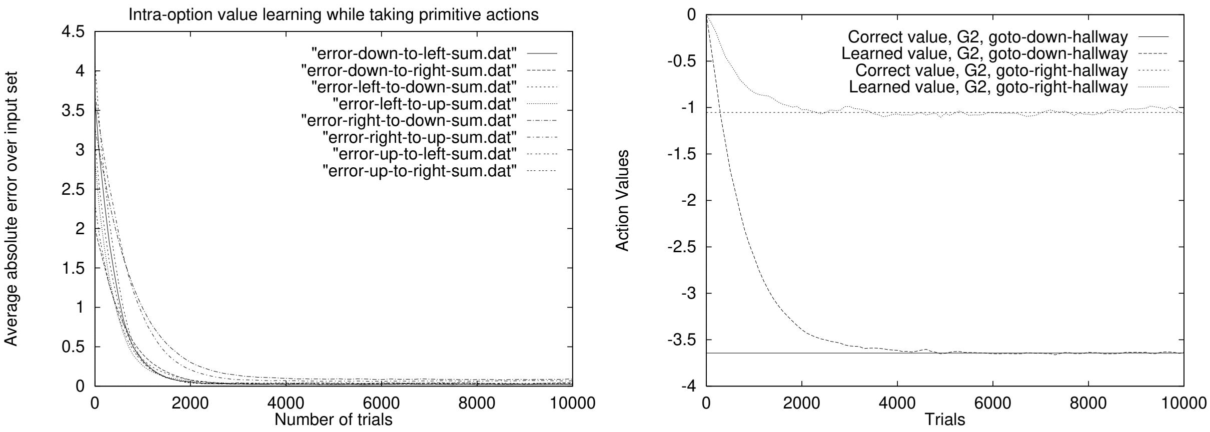

Figure 2: The learning of option values by intra-option methods without ever selecting the options. [Note: I'm polishing the figures in Adobe Illustrator now]

Q-learning, called macro Q-learning (McGovern, Sutton and Fagg, 1997). The difference between the two methods is that, when taking a multi-step option, SMDP Q-learning only updates the multi-step option, whereas macro Q-learning also updates the primitive actions that are actually executed along the way.

The learning agents start at the same states (chosen at random) and act by selecting options, according to an ϵ*-greedy* method. That is, given the current estimates $Q(s, o)$, let $o^* = \arg\max_{o \in \mathcal{O}_s} Q(s, o)$ denote the best valued action (with ties broken randomly). Then the policy used to select options was

$$\mu(s, o) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|\mathcal{O}_s|} & \text{if } o = o^* \\ \frac{\epsilon}{|\mathcal{O}_s|} & \text{otherwise,} \end{cases} \tag{15}$$

for all $s \in \mathcal{S}$ and $o \in \mathcal{O}$. The probability of a random action, ϵ, was set at 0.1 in all cases.

Figure 3 illustrates the performance of each learning algorithm, measured in two different ways. The left panel measures the average absolute error in the estimates of $Q^*_{\mathcal{O}}$ for the hallway options, averaged over the input sets $\mathcal{I}$, the eight hallway options and 30 repetitions of the whole experiment. The intra-option method shows significantly faster convergence than any of the SMDP methods. The right panel shows the quality of the greedy policy learned by each method, measured as the average value over the state space. The intra-option method is the fastest to converge to the correct optimal value.

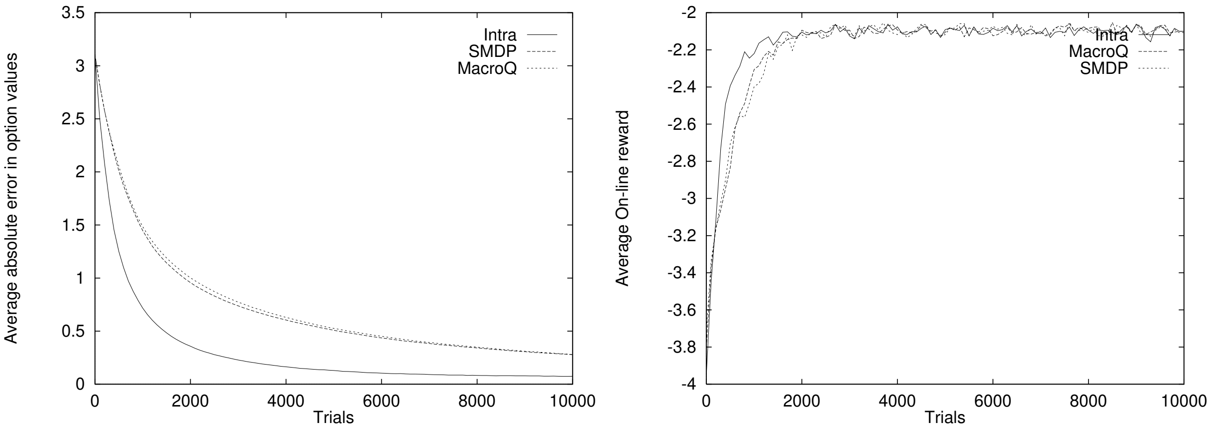

Figure 3: Comparison of SMDP, intra-option and hierarchical Q-learning. [Note: the right panel will change to an estimate of quality of the greedy policy]

7 Intra-Option Model Learning

The value of executing each option in each state can be used to determine the policy that the system is following. However, in many situations one would like to have the ability to use options during planning as well. The models of an option, r_s^o and $p_{ss'}^o$, can be learned from experience given knowledge of the option (i.e., of its π, β, and $\mathcal{I}$).

A general approach for learning the model of an option is to execute the option to termination many times in each state s, recording in each case the resultant next state s', cumulative discounted reward r, and elapsed time k. These outcomes are then averaged to approximate the expected values for r_s^o and $p_{ss'}^o$ given by (7) and (8). For example, an incremental learning rule for this could update its estimates $\hat{r}_s^o$ and $\hat{p}_{sx}^o$, for all $x \in \mathcal{S}$, after each execution of o in state s, by

$$\hat{r}_s^o = \hat{r}_s^o + \alpha[r - \hat{r}_s^o], \quad \text{and} \quad \hat{p}_{sx}^o = \hat{p}_{sx}^o + \alpha[\gamma^k \delta_{xs'} - \hat{p}_{sx}^o], \tag{16}$$

where the step-size parameter, α, may be constant or may depend on the state, option, and time. For example, if α is 1 divided by the number of times that o has been experienced in s, then these updates maintain the estimates as sample averages of the experienced outcomes. However the averaging is done, we call these *SMDP model-learning methods* because, like SMDP value-learning methods, they are based on jumping from initiation to termination of each option, ignoring what might happen along the way. In the special case in which o is a primitive action, note that SMDP model-learning methods reduce exactly to those used to learn conventional one-step models of actions (e.g., Moore and Atkeson, 1994; Sutton, 1990).

Intra-option methods for learning models were introduced by Sutton (1995) , but only for a prediction problem with a single unchanging policy, not the full control case we consider here. The idea is to use Bellman equations for the model, just as we used the Bellman equations in the case of learning value functions. The correct model of a Markov option $o =< \pi, \beta, \mathcal{I} >$ is related to itself by

$$r_s^o = \sum_{a \in \mathcal{A}_s} \pi(s,a) E\Big\{ r + \gamma(1-\beta(s')) r_{s'}^o \Big\} \quad (17)$$

$$= \sum_{a \in \mathcal{A}_s} \pi(s,a) \left[r_s^a + \sum_{s'} p_{ss'}^a (1-\beta(s')) r_{s'}^o \right], \quad (18)$$

where r and s' are the reward and next state given that action a is taken in state s, and

$$p_{sx}^o = \sum_{a \in \mathcal{A}_s} \pi(s,a) \gamma E\Big\{ (1-\beta(s')) p_{s'x}^o + \beta(s') \delta_{s'x} \Big\} \quad (19)$$

$$= \sum_{a \in \mathcal{A}_s} \pi(s,a) \sum_{s'} p_{ss'}^a (1-\beta(s')) p_{s'x}^o + \beta(s') \delta_{s'x} \quad (20)$$

for all $s, x \in \mathcal{S}$. How can we turn these Bellman equations into update rules for learning the model? First consider that action a_t is taken in s_t, and that the way it was selected is consistent with $o =< \pi, \beta, \mathcal{I} >$, that is, that a_t was selected with the distribution $\pi(s_t, \cdot)$. Then the Bellman equations above suggest the temporal-difference update rules

$$\hat{r}_{s_t}^o \leftarrow \hat{r}_{s_t}^o + \alpha \left[r_{t+1} + \gamma(1-\beta(s_{t+1})) \hat{r}_{s_{t+1}}^o - \hat{r}_{s_t}^o \right] \quad (21)$$

and

$$\hat{p}_{s_t x}^o \leftarrow \hat{p}_{s_t x}^o + \alpha \left[\gamma(1-\beta(s_{t+1})) \hat{p}_{s_{t+1} x}^o + \gamma \beta(s_{t+1}) \delta_{s_{t+1} x} - \hat{p}_{s_t x}^o \right], \quad (22)$$

where $\hat{p}_{ss'}^o$ and $\hat{r}_s^o$ are the estimates of $p_{ss'}^o$ and r_s^o, respectively, and α is a positive step-size parameter. The method we call *one-step intra-option model learning* applies these updates to every option consistent with every action taken a_t. Of course, this is just the simplest intra-option model-learning method. Others may be possible using eligibility traces and standard tricks for off-policy learning (see Sutton, 1995; Sutton and Barto, 1998).

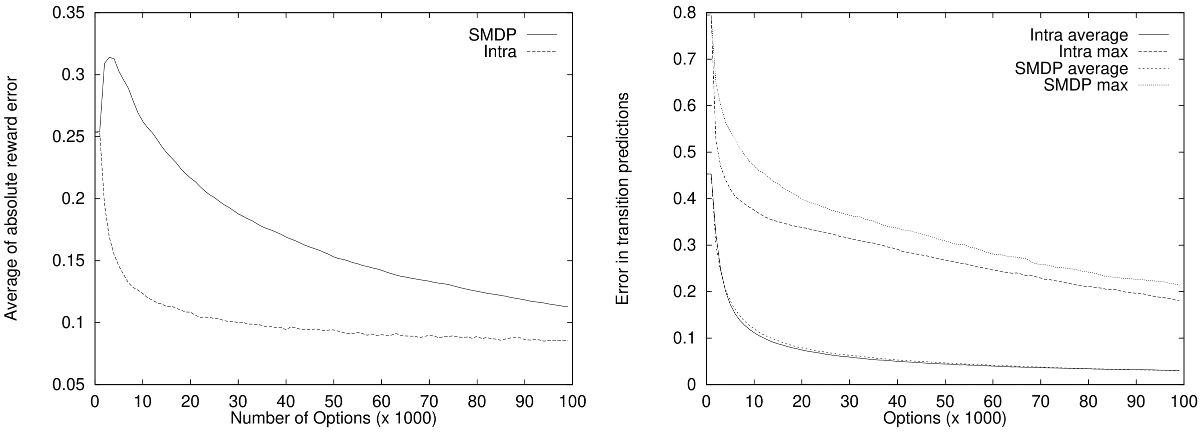

Figure 4: Learning curves for model learning by SMDP and intra-option methods. [Note: I am supposed to add 2 panel here with the graph for state S]

The intra-option learning methods should have the same advantage over SMDP methods as the ones that we have seen in the case As an illustration, consider the application of SMDP and intra-option model learning to the rooms example. As before, we assume that the eight hallway options are given, but their models are not given and must be learned. Experience is generated by selecting randomly in each state among the two possible options and four possible actions, with no goal state. In the SMDP model-learning method, equations (16) and (16) were applied whenever an option was selected, whereas, in the intra-option model-learning method, equations (21) and (22) were applied on every step to all options that were consistent with the actions taken on that step. In this example, all options are deterministic, so consistency with the action selected means simply that the option would have selected that action.

For the SMDP method, the step-size parameter was varied so that the model estimates were sample averages, which should give fastest learning. For the intra-option method, the step-size parameter was fixed, but several values were tried, $\alpha = \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$, and $\frac{1}{16}$. Figure 4 shows the learning curves for both methods. The left panel shows the average absolute error in the reward predictions, and the right panel shows the average abslute error and the maximum absolute error in the transition predictions, averaged over the eight options, and over 30 independent runs. The intra-option method provides faster convergence to correct values.

References

[Bradtke and Duff, 1995] Bradtke, S. J. and Duff, M. O. (1995). Reinforcement learning methods for continuous-time markov decision problems. In *Advances in Neural Information Processing Systems*, volume 7, pages 393–400, Cambridge, MA. MIT Press.

[Dayan, 1993] Dayan, P. (1993). Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5:613–624.

[Dayan and Hinton, 1993] Dayan, P. and Hinton, G. E. (1993). Feudal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 5, pages 271–278, San Mateo, CA. Morgan Kaufmann.

[Dietterich, 1997] Dietterich, T. G. (1997). Hierarchical reinforcement learning with the maxq value function decomposition. Technical report, Department of Computer Science, Oregon State University.

[Huber and Grupen, 1997] Huber, M. and Grupen, R. A. (1997). A feedback control structure for on-line learning tasks. *Robotics and Autonomous Systems*, 22:303–315.

[Jaakkola et al., 1994] Jaakkola, T., Jordan, M., and Singh, S. (1994). On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 6(6):1185–1201.

[Kaelbling, 1993] Kaelbling, L. P. (1993). Hierarchical learning in stochastic domains: Preliminary results. In *Proceedings of the Tenth International Conference on Machine Learning ICML'93*, pages 167–173, San Mateo, CA. Morgan Kaufmann.

[Mahadevan and Connell, 1992] Mahadevan, S. and Connell, J. (1992). Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence*, 55(2-3):311–365.

[Mahadevan et al., 1997] Mahadevan, S., Marchalleck, N., Das, T. K., and Gosavi, A. (1997). Self-improving factory simulation using continuou-time average-reward reinforcement learning. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*, pages 202–210.

[McGovern et al., 1997] McGovern, E. A., Sutton, R. S., and Fagg, A. H. (1997). Roles of macro-actions in accelerating reinforcement learning. In *Grace Hopper Celebration of Women in Computing*, pages 13–17.

[Moore and Atkeson, 1993] Moore, A. W. and Atkeson, C. G. (1993). Prioritized sweeping: Reinforcement learning with less data and less real time. *Machine Learning*, 13:103–130.

[Parr, 1998] Parr, R. (1998). Hierarchical control and learning for markov decision processes. Personnal Communication.

[Parr and Russell, 1998] Parr, R. and Russell, S. (1998). Reinforcement learning with hierarchies of machines. In *Advances in Neural Information Processing Systems*, volume 10, Cambridge, MA. MIT Press.

[Precup and Sutton, 1998] Precup, D. and Sutton, R. S. (1998). Multi-time models for temporally abstract planning. In *Advances in Neural Infomation Processing Systems*, volume 10, Cambridge, MA. MIT Press.

[Precup et al., 1998] Precup, D., Sutton, R. S., and Singh, S. (1998). Theoretical results on reinforcement learning with temporally abstract options. In *Proceedings of the Tenth European Conference on Machine Learning (ECML'98)*. Springer Verlag. In press.

[Singh, 1992] Singh, S. P. (1992). Scaling reinforcement learning by learning variable temporal resolution models. In *Proceedings of the Ninth International Conference on Machine Learning ICML'92*, pages 202–207, San Mateo, CA. Morgan Kaufmann.

[Sutton, 1990] Sutton, R. S. (1990). Integrating architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the Seventh International Conference on Machine Learning ICML'90*, pages 216–224, San Mateo, CA. Morgan Kaufmann.

[Sutton, 1995] Sutton, R. S. (1995). TD models: Modeling the world as a mixture of time scales. In *Proceedings of the Twelfth International Conference on Machine Learning ICML'95*, pages 531–539, San Mateo, CA. Morgan Kaufmann.

[Sutton and Barto, 1998] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning. An Introduction*. MIT Press, Cambridge, MA.