**On Policy Learning in Restricted Policy Spaces**

**Robby Goetschalckx** and **Jan Ramon**

Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A
B-3001 Heverlee – Belgium
robby.goetschalckx@cs.kuleuven.be
tel +32 16 32 7576

**Introduction**

We consider the problem of policy learning in a Markov Decision Process (MDP) where only a restricted, limited subset of the full policy space can be used.

A MDP consists of a state space $S$, a set of actions $A$, a transition probability function $t(s, a, s')$ and a reward function $R : S \rightarrow \mathbb{R}$. Also there is the discount factor $\gamma$. The problem is to find a policy, a mapping from states to actions $\pi : S \rightarrow A$, which gives the highest discounted return $\mathbb{E} \sum_{i=1}^{\infty} \gamma^{i+1} R(s^{(i)})$ (where $s^{(i)}$ represents the state encountered at time step $i$) for every possible start state.

However, we are not interested in any possible policy, only in a restricted, limited subset $\Pi$ of the full policy space. The assumption will be made that there exists a policy $\pi^b$ which is best for every state $s \in S$, compared to the other policies in $\Pi$. It is not required that the true optimal policy for the MDP belongs to $\Pi$.

In some settings we can also consider stochastic policies, which map states to a probability distribution over the action set. This greatly increases the size of the policy search space.

**Motivation**

There are several possible reasons for restricting the search for an optimal policy to a subset of policies $\Pi$. The full policy space has a size of the order of $O(|A|^{|S|})$. This is too large to handle in most of the interesting problem domains. For example, in relational domains, the number of states tends to be exponential in the number of objects in the world. Restricting the policy space would greatly reduce the problem complexity, especially when know that the optimal policy is in $\Pi$. For example, there might be knowledge of some form of symmetry in the state space, which implicates only policies which behave according to this same symmetry need to be considered. Domain experts might be able to tell us that certain combinations of state-to-action mappings are not useful (e.g. because they might lead to a loop). Another case is where there are external constraints causing policies outside $\Pi$ to be infeasible, so that an algorithm ignoring the constraints tends to yield optimal but unusable policies. For example, for reasons of clarity and understandability of the policy, we only want policies which can be represented as a decision tree of depth smaller than or equal to 3, or more generally, the minimal description length of the policy should be bound by some threshold to avoid overfitting.

When considering decision lists or decision trees as representations for policies, another issue might be that some of the tests are computationally expensive and the number of these in any policy should be limited.

Consider the following example of a problem where there is a restricted policy space. The MDP represents a temperature control system with following actions: turn on a heating system, turn on a cooling system, turn the systems off. In such a system it is intuitively clear that a good policy will heat at low temperatures, will at some threshold temperature $t_1$ turn off the systems, and at another threshold temperature $t_2 > t_1$ turn on some cooling system. What the threshold temperatures are and which of various heating and cooling systems will be turned on is not clear, but the policy space to be explored is greatly reduced in size.

The traditional reinforcement learning techniques could be used to learn the optimal policy in most of these settings, but this policy might possibly not fit the constraints. It might also take a lot of time exploring parts of the policy space which are not valid according to the restrictions. If the restrictions on the policy space are strict, intuitively it seems that a better performance could be gained, as exploration will be limited to a small subset of the full policy space. It is, however, not trivial to find a general approach for this setting.

**Related Work**

The related work in policy search can be roughly divided in two groups. On the one hand, there is the policy search in the full policy space (either deterministic policies or stochastic policies). This search can be done with evolutionary algorithms (Moriarty, Schultz, & Grefenstette 1999), a gradient ascent approach (Madani 2002; Kakade 2002), or using pairwise comparisons for an implicit gradient ascent which does not require a model of the MDP (Strens & Moore 2002). The basic difficulty for using these approaches in restricted spaces is that these approaches are not designed to handle constraints on the policy space. The difference with what we are interested in is that we will use background knowledge and other types of constraints to restrict the policy search space, which should speed up the process.

Copyright © 2007, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.
On the other hand, there are the experts approaches which handle a limited number of policies, called experts. Here every policy is observed for some time (a multiple of the mixing time of the resulting Markov chain, the time needed to reach the stable state probability distribution of the chain) and the performance of the policies is used to decide which is best, which reduces the problem to the typical bandit-setting. These settings are generally easier, and theoretic bounds on performance, both in asymptotical return and regret, can be found in the literature (Talvitie & Singh 2007; de Farias & Megiddo 2004; Herbster & Warmuth 1995). The main difference with the restricted policy space setting is the size of the policy space. We are interested in methods which could handle large, possible infinite subsets of the policy space. Here it is infeasible in general to try every possible policy for enough steps to be able to estimate its value accurately.

Possible Approaches and Difficulties

A first approach which should be considered is the use of the policy gradient ascent algorithms combined with the constraints on the policy space. For these approaches in most cases an accurate model of the MDP is needed (or a simulator or another way to generate runs in the MDP using fixed start states and a fixed sequence of random numbers, as in (Strens & Moore 2002)). This does however seem most promising, especially if a method could be found which avoids the need for a complete model of the MDP.

Another approach, which can be taken if the number of policies is finite (or possibly enumerable), the state space is limited and an accurate model is given, and if we can make the assumption that there exists a best policy \( \pi^b \), is the following. Approximate the value function in every state with a dynamic programming approach. With this approach we can calculate upper and lower bounds on the value of every state using two different policies \( \pi_1 \) and \( \pi_2 \). These bounds become more and more strict with each iteration. If we know that using some policy \( \pi_1 \) a state will have a lower value than another policy \( \pi_2 \), then \( \pi_1 \) will no longer be considered. This approach has a number of drawbacks, however: the state space should not be too large, and evaluating a policy takes an amount of time of the order of the mixing time of the policy.

A third approach is keeping a value for every policy, and perform random exploration, at every step updating the values of the policies which concurred with the taken action. This can be proven to work in the limit for MDP’s where the assumption of the existence of a best policy is even stricter:
\[
\exists \pi^b \in \Pi : \forall \pi \in \Pi, \forall s \in S : Q^\pi(s, \pi^b(s)) \geq Q^\pi(s, \pi(s)).
\]
This means that, for every policy \( \pi \) in the restricted space, performing the best policy for one step, and afterwards following \( \pi \) will give a better result than following \( \pi \) right away. This assumption is too strict to make this approach applicable in most domains. Also, the number of policies in the restricted space must be finite. Even then it will take much more time than the mixing time to be able to discern the better policy.

If a method is found which works for the restricted policy space, it could possibly be generalized by considering a policy preference function, where the value of a policy is multiplied by its preference value. The restricted policy setting would then translate to using a policy preference function mapping all policies in \( \Pi \) to 1 and all other policies to 0.

The assumption of the existence of a best policy \( \pi^b \) might be too strict for a large number of interesting application domains. Another way to define the best policy as the policy which has the best asymptotic value, this is the value function of all states, weighted by the stable state probability function of the Markov chain. Another approach is weighing by the distribution of possible start states, which is a more natural approach in the case of episodal domains. On the other hand, if we do have the information that such a best policy exists, this could be used as extra information, which speeds up the learning process. This is the case in the second approach discussed in this section.

Conclusion

Policy search in a restricted policy space is an interesting concept which has not yet been investigated. In a lot of non-trivial domains the policy space can be restricted with the use of various kinds of background knowledge. The most promising approach seems to be an adaptation of the existing policy search techniques to this new setting. This adaptation does not seem trivial, however, and there are various levels of difficulty to be considered.

References


