Using MDP Characteristics to Guide Exploration in Reinforcement Learning

Bohdana Ratitch and Doina Precup

McGill University, Montreal, Canada
{bohdana,dprecup}@cs.mcgill.ca,
http://www.cs.mcgill.ca/{~sonce,~dprecup}

Abstract. We present a new approach for exploration in Reinforcement Learning (RL) based on certain properties of the Markov Decision Processes (MDP). Our strategy facilitates a more uniform visitation of the state space, a more extensive sampling of actions with potentially high variance of the action-value function estimates, and encourages the RL agent to focus on states where it has most control over the outcomes of its actions. Our exploration strategy can be used in combination with other existing exploration techniques, and we experimentally demonstrate that it can improve the performance of both undirected and directed exploration methods. In contrast to other directed methods, the exploration-relevant information can be precomputed beforehand and then used during learning without additional computation cost.

1 Introduction

One of the key features of reinforcement learning (RL) is that a learning agent is not instructed what actions it should perform; instead, the agent has to evaluate all available actions [13], and then decide for itself on the best way of behaving. This creates the need for an RL agent to actively explore its environment, in order to discover good behavior strategies. Ensuring an efficient exploration process and balancing the risk of taking exploratory actions with the benefit of information gathering are of great practical importance for RL agents, and have been the topic of much recent research, e.g., [14, 7, 2, 4, 12].

Existing exploration strategies can be divided into two broad classes: undirected and directed methods. *Undirected methods* are concerned only with ensuring *sufficient* exploration, by selecting all actions infinitely often. The ε-greedy and Boltzman exploration strategies are notable examples of such methods. Undirected methods are very popular because of their simplicity, and because they do not have additional requirements of storage or computation. However, they can be very inefficient for certain domains. For example, in deterministic goal directed tasks with a positive reward received only upon entering the goal state, undirected exploration is exponential in the number of steps needed for an optimal agent to reach the goal state [14]. On the other hand, by using some information about the course of learning, the same tasks can be solved in time polynomial in the number of states and maximum number of actions available in each state [14]. The impact of exploration is believed to be even more important for stochastic environments. *Directed exploration* strategies attempt not only to ensure a

N. Lavrač et al. (Eds.): ECML 2003, LNAI 2837, pp. 313–324, 2003.

sufficient amount of exploration, but also to make the exploration *efficient*, by using additional information about the learning process. These techniques often aim to achieve a more uniform exploration of the state space, or to balance the relative profit of discovering new information versus exploiting current knowledge. Typically, directed methods keep track of information regarding the learning process and/or learn a model of the system. This requires extra computation and storage in addition to the resources needed by general on-line RL algorithms, in order to make better exploration decisions. More details on existing exploration methods are given in Section 2.2.

In this paper, we present a new directed exploration approach, which takes into account the properties of the Markov Decision Process (MDP) being solved. In prior work [9], we introduced several attributes that can be used to provide a quantitative characterization of MDPs. Our approach to exploration is based on the use of the two attributes: state transition entropy and forward controllability. The state transition entropy provides a characterization of the amount of stochasticity in the environment. Forward controllability measures how much the agent's actions actually impact the trajectories that the agent follows. Our prior experimental results [9] suggest that these attributes significantly affect the quality of learning for on-line RL algorithms with function approximation, and that this effect is due in part to the amount of exploration in MDPs with different characteristics. In this paper, we show how to use these MDP attributes in combination with both undirected and directed existing exploration methods.

Using MDP attributes can improve the exploration process in three ways. First, it encourages a more homogeneous visitation of the state space, similar to other existing directed methods. Second, it encourages more frequent sampling for actions with potentially high variance in their action-value estimates. Finally, it encourages the learning agent to focus more on the states in which its actions have more impact. One important difference between our exploration strategy and other directed techniques is that the extra information we use reflects only properties of the task at hand, and does not depend on the history of learning. Hence, this information does not carry the bias of previous, possibly unfortunate exploration decisions. Additionally, in some cases the MDP attributes can be pre-computed beforehand and then used during learning without any additional computational cost. The attributes' values can also be transferred between tasks if the agent is faced with solving multiple related tasks in an environment in which the dynamics does not change much. The attributes can also be estimated during learning, which would require only a small constant amount of additional resources in contrast to most other directed methods.

The rest of the paper is organized as follows. In Section 2, we provide background on RL and existing exploration approaches. The details of the proposed exploration method are presented in Section 3. Empirical results are discussed in Section 4. The directions for future work are presented in Section 5.

2 Background

2.1 RL Framework

We assume the standard RL framework, in which a learning agent is situated in a dynamic stochastic environment and interacts with it at discrete time steps. The envi-

ronment assumes states from some *state space* S and the agent chooses actions from some *action space* A. On each time step, in response to the agent's actions, the environment undergoes state transitions governed by a stationary probability distribution $P^a_{ss'}$, where $s, s' \in S, a \in A$. At the same time, the agent receives a numerical *reward* from the environment, $R^a_{ss'} \in \Re$, which reflects the one-step desirability of the agent's actions. State transitions and rewards are, in general, stochastic and satisfy the *Markov property*: their distributions depend only on the current state of the environment and the agent's current action and are independent of the past history of interaction. The goal of the agent is to adopt a *policy* (a way of choosing actions) $\pi : S \times A \rightarrow [0,1]$ that optimizes a *long-term performance* criterion, called *return*, which is usually expressed as a cumulative function of the rewards received on successive time steps. Such a learning problem is called a Markov Decision Process (MDP). Many RL algorithms estimate *value functions* which can be viewed as utilities of states and actions. Value functions are estimates of the expected returns and take into account any uncertainty pertaining to the environment or the agent's action choices. For instance, the *action-value function* associated with a policy π, $Q^\pi : S \times A \rightarrow \Re$, is defined as:

$$Q^\pi(s,a) = E_\pi \{r_{t+1} + \gamma r_{t+2} + \ldots | s_t = s, a_t = a\}$$

where $\gamma \in (0,1]$ is the discount factor. The optimal action value function, Q^*, is defined as the action-value function of the best policy: $Q^*(s,a) = \max_\pi Q^\pi(s,a)$. In this paper, we focus on RL algorithms that estimate the optimal action-value function from samples obtained by interacting with the environment.

2.2 Exploration in RL

The goal of an exploration policy is to allow the RL agent to gather experience with the environment in such a way as to find the optimal policy as quickly as possible, while also gathering as much reward as possible during learning. This goal can be itself cast a learning problem, often called *optimal learning* [6]. Solving this problem would require the agent to have a probabilistic model of the uncertainty about its own knowledge of the environment, and to update this model as learning progresses. Solving the optimal learning problem then becomes equivalent to solving the partially observable MDP (POMDP) defined by this model, which is generally intractable. However, various heuristics can be used to decide which exploration policy to follow, based only on certain aspects of the uncertainty about the agent's knowledge of the environment.

As discussed in Section 1, existing exploration techniques can be grouped in two main categories: undirected and directed methods. Undirected methods ensure that each action will be selected with non-zero probability in each visited state. For instance, the ε-*greedy exploration strategy* selects the currently greedy action (the best according to the current estimate of the optimal action-value function $Q(s,a)$), in any given state, with probability $(1-\varepsilon)$, and selects a uniformly random action with probability ε. Another popular choice for undirected exploration, the *Boltzman distribution* assigns probability $\pi(s,a)$ of taking action a in state s as $\pi(s,a) = e^{\frac{Q(s,a)}{\tau}} / \sum_{b \in A} e^{\frac{Q(s,b)}{\tau}}$, where τ is a positive temperature parameter that decreases the amount of randomness as it approaches zero. When using on-policy RL algorithms, such as SARSA [13], the exploration rate (ε in the ε-greedy exploration and τ in Boltzman exploration) has to decrease

to zero with time in an appropriate manner [11] in order to ensure convergence to the optimal (deterministic) policy. In practice, however, constant exploration rates are often used.

Directed exploration methods typically keep some information about the state of knowledge of the agent, estimating certain aspects of its uncertainty. The action to be taken is usually selected by maximizing an evaluation function that combines action-values with some kind of *exploration bonuses*, δ_i:

$$N(s,a) = K_0 Q(s,a) + K_1 \delta_1(s,a) + \ldots + K_k \delta_k(s,a) \quad (1)$$

Exploration is driven mainly by the exploration bonuses that change over time. The positive constants K_i control the exploration-exploitation balance.

Directed exploration methods differ in the kind of exploration bonuses they define, which reflect different heuristics regarding what states are important to revisit. For example, counter-based methods [14] direct exploration toward the states that were visited least frequently in the past. Recency-based exploration [14, 12] prefers instead the states that were visited least recently. In both of these cases, the result is a more homogeneous exploration of the state space. Error-based exploration [10] prefers actions leading to states whose value changed most in past updates. Interval Estimation (IE) [3, 16], as well as its global equivalent, IEQL+ [7], bias exploration toward actions that have the highest variance in the action value samples. In the value of information strategy [1, 2], the exploration-exploitation tradeoff is solved with a myopic approximation of the value of perfect information. The E^3 algorithm [4] learns a model of the MDP. Based on the estimated accuracy of this model and a priori knowledge of the worst-case mixing time of the MDP and the maximum attainable returns, E^3 explicitly balances the profit of exploitation and the possibility of efficient exploration. Due to this balancing, E^3 provably achieves near-optimal performance in polynomial time. However, there is little practical experience available with this algorithm.

3 Using MDP Attributes for Exploration

Similarly to many directed exploration methods, the goal of our approach is to ensure a more uniform visitation of the state space, while also gathering quickly the samples most needed to estimate well the action value function. In order to achieve this goal, we focus on using two attributes that can be used to characterize MDPs: state transition entropy (STE) and forward controllability (FC). In prior work [9], we found that these attributes had a significant effect on the speed of learning and quality of the solution found by on-line RL algorithms. This effect seemed to be due mostly to their influence on the RL agent's exploration of the state space. Both attributes can be computed for each state-action pair (s,a) based on the MDP model (if it is known) or they can be estimated based on sample transitions. The basic idea of our strategy is to favor exploratory actions which exhibit high values of STE, FC, or both of these features. We will now explain the details of our approach.

State transition entropy (STE) measures the amount of stochasticity due to the environment's state dynamics. Let $O_{s,a} \in S$ denote a random variable representing the outcome (next state) of the transition from state s when the agent performs action a.

Using the standard information-theoretic definition of entropy, STE for a state-action pair (s,a) can be computed as follows [5]:

$$STE(s,a) = H(O_{s,a}) = -\sum_{s' \in S} P^a_{s,s'} \log P^a_{s,s'} \tag{2}$$

A high value of $STE(s,a)$ means that there are many possible next states s' (with $P^a_{s,s'} \neq 0$) which occur with similar probabilities. If in some state s, actions a_1 and a_2 are such that $STE(s,a_1) > STE(s,a_2)$, the agent is more likely to encounter more different states by taking action a_1 than by taking action a_2. This means that giving preference to actions with higher STE could achieve a more homogeneous exploration of the state space. Empirical evidence that a homogeneous visitation of the state space can be helpful is present in [13], where the performance of Q-learning with an ε-greedy behavior policy is compared with the performance of Q-learning performed by picking states uniformly randomly. The experiments were performed on discrete random MDPs with different branching factors. Note that a large branching factor means a high STE value for all states. In these tasks, the ε-greedy on-policy updates resulted in better solutions and faster learning mainly for the deterministic tasks (with branching factor 1). As the branching factor (and thus STE) increased, performing action-value updates uniformly across the state space led to better solutions in the long run, and to better learning speed.

Another potential consequence of a high value of $STE(s,a)$ is a large variance of the action-value estimates for (s,a). In on-policy learning methods, such as SARSA [13], the action value of a state-action pair (s,a) is updated toward a *target estimate* obtained after taking action a:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha[\underbrace{R^a_{ss'} + \gamma Q(s',a')}_{\text{Target for } (s,a)}], \alpha \in (0,1)$$

These target estimates are drawn according to the probability distribution of the next state s'. If $STE(s,a)$ is high, there will be many possible next states, and consequently the variance in the target estimates could be higher. In prior experiments using the SARSA(0) learning algorithm with linear function approximation [9], we observed that in environments with high STE values, there was a trade-off in the quality of the approximation achieved between the positive effect of "natural" exploration and the negative effect of high variance in the target action-value estimates used by the algorithm. In order to get a good estimate of $Q(s,a)$ when the target values have high variance, more samples are needed. By encouraging the exploration of actions with high STE values, our strategy ensures that we will collect enough samples. This idea is reminiscent of the IE directed exploration method [3], but we do not rely on explicitly estimating the variance of the action value samples, which would be much more expensive in terms of both storage and computation.

The *controllability* of a state s is a normalized measure of the information gain when predicting the next state based on knowledge of the action taken, as opposed to making the prediction before an action is chosen (a similar, but not identical, attribute is used by Kirman [5]). Let $O_s \in S$ denote a random variable representing the outcome of a uniformly random action in state s. Let $O_s \in S$ denote a random variable representing the outcome of a uniformly random action in state s. Let A_s denote a random variable

representing the action taken in state s. We consider A_s to be chosen from a uniform distribution. Given the value of A_s, information gain is the reduction in the entropy of O_s: $H(O_s) - H(O_s|A_s)$, where

$$H(O_s) = -\sum_{s' \in S}\left(\frac{\sum_{a \in A} P^a_{s,s'}}{|A|}\right)\log\left(\frac{\sum_{a \in A} P^a_{s,s'}}{|A|}\right); \quad H(O_s|A_s) = -\sum_{a \in A}\frac{1}{|A|}\sum_{s' \in S} P^a_{s,s'}\log(P^a_{s,s'})$$

The controllability in state s is defined as:

$$C(s) = \frac{H(O_s) - H(O_s|A_s)}{H(O_s)} \tag{3}$$

If all actions are deterministic, then $H(O_s|A_s) = 0$ and $C(s) = 1$. If $H(O_s) = 0$ (all actions deterministically lead to the same state), then $C(s)$ is defined to be 0. The *forward controllability* (FC) of a state-action pair is the expected controllability of the next state:

$$FC(s,a) = \sum_{s' \in S} P^a_{s,s'} C(s') \tag{4}$$

Favoring actions with high FC will direct an RL agent toward states in which it has a lot of control on the next state transitions, by making appropriate action choices. Having such control enables the agent to reap higher returns in environments where some trajectories are more profitable than others, as shown in our prior experiments [9]. At the same time, actions with high FC lead to states in which different actions have very different outcomes. Hence, from such states, the agent is likely to explore the state space more uniformly. A third reason to favor actions with high values of $FC(s,a)$ is that, similarly to the case of high STE, such actions can potentially have high variance in the targets used to update their action values, $Q(s,a)$. If a resulting state, s', is highly controllable, the actions a' available there could lead to very different next states, and hence $Q(s',a')$ is likely to have high variance. As a result, gathering more samples from (s,a) should increase the speed of learning.

The idea of guiding exploration based on the values of the STE and FC attributes can easily be incorporated in both undirected and directed exploration techniques. For instance, consider the case of the ε-greedy exploration strategy. The greedy action is still chosen with probability $(1-\varepsilon)$. When a choice to explore is made (with probability ε), the exploratory action is selected according to a Boltzman distribution:

$$\pi(s,a) = \frac{e^{\frac{K_1 * STE(s,a) + K_2 * FC(s,a)}{\tau}}}{\sum_{b \in A} e^{\frac{K_1 * STE(s,b) + K_2 * FC(s,b)}{\tau}}} \tag{5}$$

where τ is the temperature parameter. The nonnegative constants K_1 and K_2 can be used to adjust the relative contribution of each term. Of course, STE and FC can be used with probability distributions other than Boltzman as well.

In directed exploration, the STE and FC attributes can be used as additional exploration bonuses, and hence can be easily incorporated in most existing methods. In this case, the behavior policy deterministically picks the action maximizing the function:

$$N(s,a) = K_0 Q(s,a) + K_1 STE(s,a) + K_2 FC(s,a) + \sum_j K_j \delta_j(s,a) \tag{6}$$

where $\delta_j(s,a)$ can be any exploration bonuses based on data about the learning process, such as counter-based, recency-based, error-based or IE-based bonuses. In this case, the trade-off between exploitation and exploration can be controlled by tuning the parameters K_i associated with each term.

Note that our exploration approach uses only characteristics of the environment, which are *independent of the learning process*. Thus, the information needed can be gathered prior to learning. This can be done if the transition model is known, or if the agent has an access to a simulator, with which it can interact to estimate the attributes from sampled state transitions[1]. Also, the attributes' values can be carried over if the task changes slightly (e.g., in the case of goal-directed tasks in which the goal location moves over time). Alternatively, the attributes can be computed during learning based on observed state transitions. This can be done efficiently by incremental methods for entropy estimation [15] and mean estimation with a forgetting factor for FC. In this case, only a small constant amount of extra computation per time step is needed. This is in contrast to most other directed exploration methods, which not only rely on estimation of transition probabilities, but also require more computation to re-evaluate their exploration-relevant information on every time step, e.g., [14, 4, 12, 3, 16, 2]. At the same time, the exploration-relevant information based on the learning history used in other directed techniques can carry the bias of previous (possibly unsuccessful) exploration decisions and value estimates.

4 Experimental Results

In order to assess empirically the merit of using STE and FC as heuristics for guiding exploration, we experimented with using these attributes together with ε-greedy exploration (as a representative of undirected methods) and recency-based exploration (as a representative of directed methods). We chose recency-based exploration among the directed exploration techniques because in previous experiments [14] it compared favorably to other directed methods, while being less sensitive to the tuning of its parameters. At the same time, this method is conceptually close to attribute-based exploration, in that it encourages a homogeneous exploration of the state space. Hence, it is interesting to see whether the use of MDP attributes can give any additional benefit in this case.

The attributes were incorporated into the ε-greedy strategy as shown in (5). We used parameter settings $K_1, K_2 \in \{0,1\}$, $\tau = 1$ and $\varepsilon \in \{0.1, 0.4, 0.9\}$. The recency-based technique was combined with the attributes based on the idea of additive exploration bonuses, as shown in (6), where we used one recency-based exploration bonus, $\delta(s,a)$. As before, we used $K_1, K_2 \in \{0,1\}$. The constant corresponding to the value function was set to $K_0 \in \{1, 10, 50\}$ and the constant corresponding to the recency bonus was $K_3 = 1$. The learning algorithm used was tabular SARSA(0) with a decreasing learning rate $\alpha(s_t,a_t) = \frac{1.25}{0.5+n(s_t,a_t)}$, where $n(s_t,a_t)$ is the number of visits to a state-action pair (s_t,a_t) at time t. The action values $Q(s,a)$ were initialized to zero at the beginning of learning.

[1] Note that even if the MDP model is known, it is often not feasible to apply dynamic programming methods and the issue of efficient exploration is still important. As suggested in [17], model-based exploration methods are in fact superior to model-free methods in many cases.

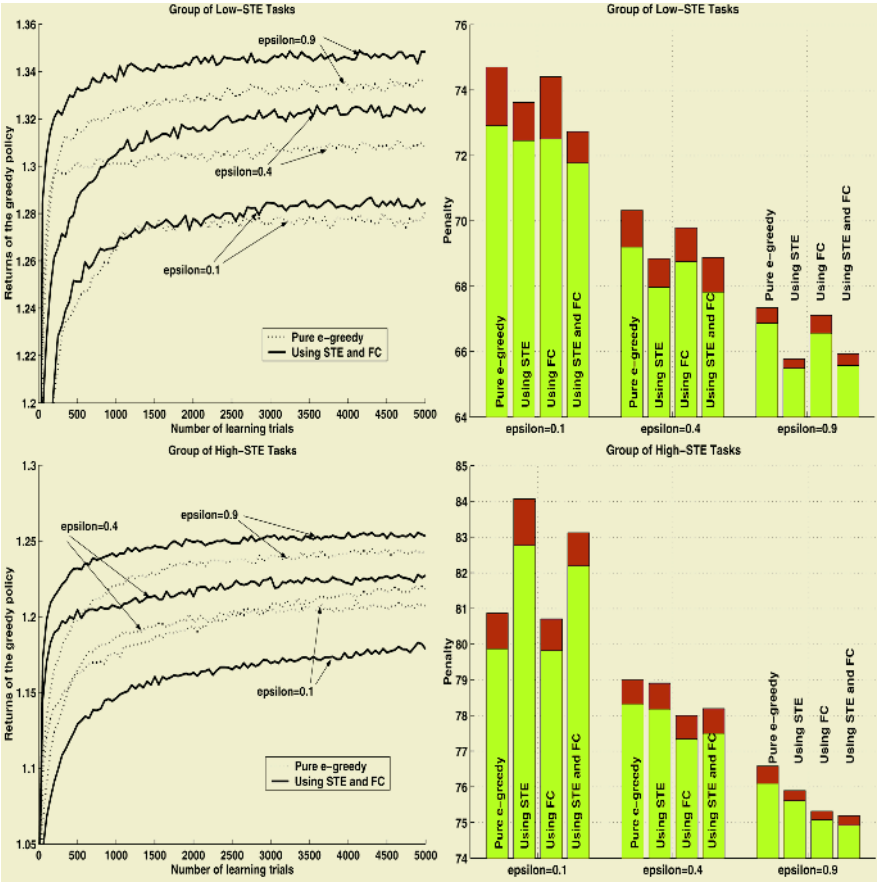

Fig. 1. Performance of ε-greedy exploration (pure and attribute-based) for low-STE (top) and high-STE tasks (bottom)

The experiments were conducted on randomly generated MDPs with 225 states and 3 actions available in every state. The branching factor for these MDPs varied randomly between 1 and 20 across the states and actions. Transition probabilities and rewards were also randomly generated, with rewards drawn uniformly from $[0,1]$. At each state, there was a 0.01 probability of terminating the episode. These random MDPs were divided in four groups of five tasks each. Two of the groups contained MDPs with "low" average STE values ($\text{avg}(STE(s,a)) < 1.7$), and the other two groups contained MDPs with "high" STE values ($\text{avg}(STE(s,a)) \in [1.7, 2.7]$). This grouping allowed us to investigate whether the overall amount of stochasticity in the environment influences the effect of the attributes on exploration. The two groups on each STE level differed in that one of them (which we will call the *test group*) had a large variation in the attribute values for different actions, while the other one (the *control group*) had similar values of the attributes across all states and actions. In the control groups, we would expect to see no effect of using the attributes, because the exploration decisions at all states and

actions should be mostly unaffected by the attribute values. Hence, we use the control groups to test the possibility of observing any effect "by chance". The experimental results presented below are for the case, where the attributes where precomputed from simulation of the MDPs prior to learning. Preliminary experiments, where the attributes were computed during learning, indicate qualitatively similar results.

We use two measures of performance for the exploration algorithms under consideration. The first measure is an estimate of the return of the greedy policy produced by the algorithms at different points in time. After every 50 trials, we take the greedy policy with respect to the current action values and we run this policy from 50 fixed test states, uniformly sampled from the state space. We run 30 trials from each such state, then we average these returns over the trials and over the 50 states. Because we are using different tasks, with different optimal value functions (and hence different upper bounds on the performance that can be achieved), it is difficult to compare greedy returns directly, without any normalization. Hence, we normalize the average greedy return by the average return of the uniformly random policy from the same 50 states (computed over 30 trials). In our prior experiments [9] we found that this normalization yields very similar results to normalizing by the return of the optimal policy. Of course, using the optimal policy would generally give the best normalization, but the optimal policy cannot always be computed by independent means.

The second performance measure that we use is aimed at providing a quantitative measure of both the speed of learning and the quality of the solution obtained. It is often difficult to compare different algorithms in terms of both of these measures, because one algorithm may have a steeper learning curve, but a more erratic (or worse) performance in the long run. In order to assess these kinds of differences, we use the following penalty measure for each run:

$$P = \sum_{t=1}^{T} \frac{t}{T}(R_{max} - R^t), \tag{7}$$

where R_{max} is an upper limit of the (normalized) return of the optimal policy[2], R^t is the (normalized) greedy return after trial t and T is the number of trials. In this way, failure to achieve the best known performance is penalized more after more learning trials have occurred. This measure gives a lower penalty to methods that achieve good solutions earlier and do not deviate from them. In our experiments, we compute one penalty for every independent run of every algorithm (which can be viewed as a "summary" for the run).

The results of the experiments are presented in Figure 1, for the ε-greedy strategy, and in Figure 2, for recency-based exploration. The performance measures are computed in terms of the normalized greedy returns, averaged over the 5 MDPs in each group and over 30 runs for each MDP. The left panels represent learning curves for the normalized greedy returns, while the right panels represent the average penalty measure over the runs, computed using (7). Light lower portions of the bars represent mean penalty, and they are topped with standard deviation (dark portions).

We also performed statistical tests to verify whether the observed performance differences are statistically significant. Because we are interested in both the asymptotic

[2] This limit can be either known or estimated as a maximum return ever observed for a task.

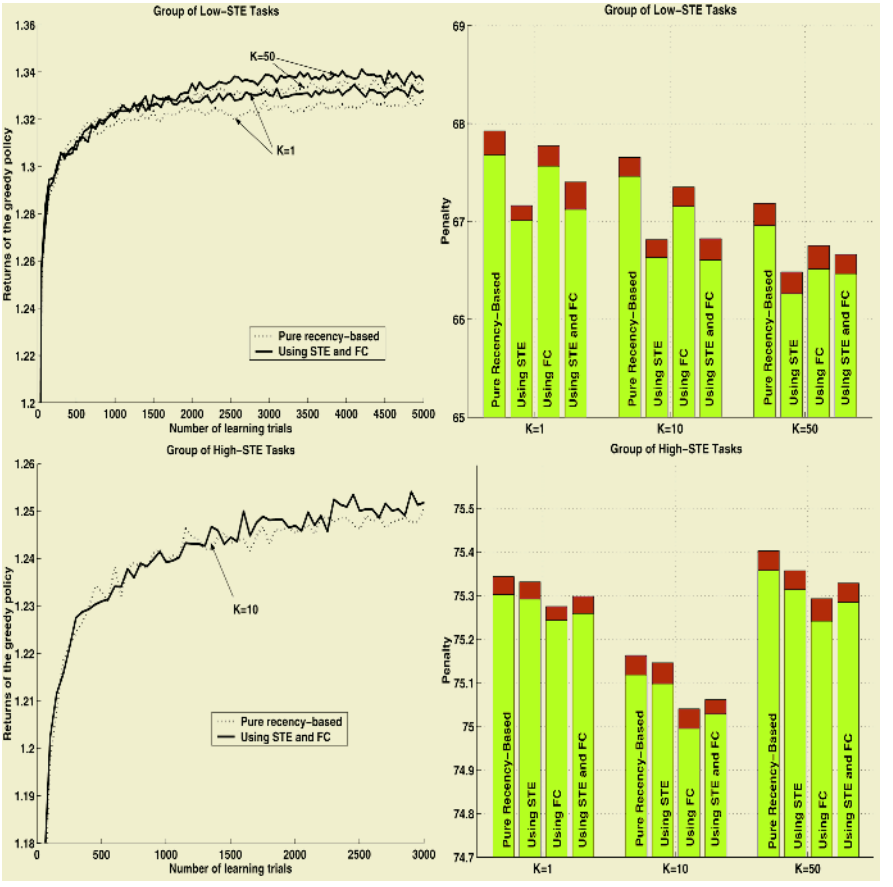

Fig. 2. Performance of pure and attribute-based recency exploration for the low-STE (top) and high-STE tasks (bottom)

performance and the speed of learning, we used a randomized ANOVA procedure [8] to compare the learning curves of the different algorithms. This procedure is more appropriate than the conventional one for comparing learning curves, because it does not rely on the assumption of homogeneity of co-variance, which is violated when there are carry-over effects. We performed the analysis separately on the learning curves for each task. We also performed two-way ANOVA of the penalty measure averaged over the 5 MDPs in each group. In this case, one factor was the tunable parameter for the "pure" exploration strategy (ε for the ε-greedy and K for the recency-based) and the other factor was the variant of the corresponding strategy (pure vs. using the attribute(s)).

As shown in Figure 1, incorporating the attributes into the ε-greedy strategy has a positive effect both for the low-STE and for the high-STE test, in all cases except $\varepsilon = 0.1$ in high-STE environments. The randomized ANOVA test for learning curves showed a difference in the performance between the pure strategy and each of the three attribute-based variants at the level of significance no smaller than $p = 0.008$ for each task and

for each setting of ε. The penalty measure graphs show that the positive effect of using STE becomes more significant as ε increases, and this trend is especially pronounced in the case of the high-STE tasks. In this case, most estimates $Q(s,a)$ have high variance, but with a small exploration rate, many actions are not sufficiently sampled. Using STE allows more samples to be gathered for such actions, and hence improve the solution quality. FC has a greater positive effect for the high-STE tasks, as can be seen from the penalty graphs in the right panels of Fig.1, mainly because it improves the speed of learning (the learning curves are not shown here, due to lack of space). This shows that encouraging the agent to learn about states where it can better control the course of state transitions is helpful especially given a background of high overall stochasticity. The two-way ANOVA test on the penalty measure showed that the positive effect of using the attributes is significant.

Figure 2 presents the same performance measures when incorporating the attributes in the recency-based exploration strategy. The recency-based method is significantly more robust to the tuning of its main parameter, K_0 than the ε-greedy strategy is to the tuning of ε. With all the settings of K_0, the performance of this strategy is close to the best performance of the ε-greedy strategy (obtained at $\varepsilon = 0.9$). However, using the MDP attributes further improves performance of the recency-based method as well, although the effects appear to be smaller than in the case of the ε-greedy method (we believe this is due to a ceiling effect). Although the differences appear to be small, the statistical tests show that most differences are significant. In particular, the randomized ANOVA test shows a significant difference between learning curves in the low-STE group at the level no smaller than $p = 0.04$ for all tasks and all attribute versions. For the high-STE tasks, significance levels range from $p = 0.04$ for the version using only FC to $p = 0.226$ for the version using only STE. The two-way ANOVA on the penalty measure is also less significant for the recency-based strategy in the high-STE group ($p = 0.03$ for comparison of the pure vs. FC-based variant and $p = 0.11$ for pure vs. STE-based variant). Similar to the case of the ε-greedy strategy, FC appears to have a greater positive effect for the high-STE tasks.

For both the ε-greedy and recency-based strategies, in most cases, using STE and FC together produces an improvement which is very similar to the best improvement obtained by using either one of the attributes in isolation. For the low-STE test group, the STE attribute brings a bigger performance improvement, whereas for the high-STE test group, FC has a bigger effect. Thus, it would be reasonable to always use the combination of two attributes to achieve the best improvement. Note that the improvements were obtained without tuning any additional parameters, both for the ε-greedy and the recency-based methods.

The results of the experiments conducted on the control groups did not reveal any effect of using the attributes with either the ε-greedy or recency-based exploration. This reinforces our conclusion that the effects observed on the test groups are not spurious.

5 Conclusions and Future Work

In this paper, we introduced a novel exploration approach based on the use of specific MDP characteristics. Exploration decisions are made independently of the course of

learning so far, based only on properties of the environment. Our technique facilitates a more homogeneous exploration of the state space, a more extensive sampling of actions with a potentially high variance of the action-value function estimates and encourages the agent to focus on states where it has most control over the outcomes of its actions. In our experiments, using these attributes improved performance for both undirected (ε-greedy) and directed (recency-based) exploration in a statistically significant way. The improvements were obtained without tuning any additional parameters. The attribute values can be pre-computed before the learning starts, or they can be estimated during learning. In the latter case, the amount of additional storage and computation is much less compared to other directed techniques.

We are currently conducting a more detailed empirical study using toy hand-crafted MDPs in order to better understand the circumstances under which the use of MDP attributes to guide exploration is most beneficial. We also plan to investigate the use of other attributes, e.g. the risk of taking exploratory actions and variance of immediate rewards.

References

1. Dearden, R., Friedman, N., Russell, S.: Bayesian Q-learning. In Proc. AAAI (1998) 761-768
2. Dearden, R., Friedman, N., Andre, D.: Model-Based Bayesian Exploration. In Proc. of the 15th UAI Conference (1999) 150-159
3. Kaelbling, L.P.: Learning in embedded systems (1993) Cambridge, MIT Press
4. Kearns, M., Singh, S.: Near-Optimal Reinforcement Learning in Polynomial Time. In Proc. of the 15th ICML (1998) 260-268
5. Kirman, J.: Predicting Real-Time Planner Performance by Domain Characterization. Ph.D. Thesis, Brown University (1995)
6. Kumar, P.R.: A survey of some results in stochastic adaptive control. SIAM Journal of Control and Optimization **23** (1985) 329-338
7. Meuleau, N., Bourgine, P.: Exploration of Multi-State Environments: Local Measures and Back-Propagation of Uncertainty. Machine Learning **35**(2) (1999) 117-154
8. Piater, J.H., Cohen, P.R., Zhang,X., Atighetchi, M.: A Randomized ANOVA Procedure for Comparing Performance Curves. In Proc. of the 15th ICML (1998) 430-438
9. Ratitch, B., Precup,D.: Characterizing Markov Decision Processes. In Proc. of the 13th ECML (2002) 391-404
10. Schmidhuber, J.H.: Adaptive Confidence and Adaptive Curiosity. Technical Report FKI-149-91, Technische Universitat Munchen (1991)
11. Singh, S., Jaakkola, T., Littman, M.L.,Szepesvari, C.: Convergence Results for Single-Step On-Policy Reinforcement Learning Algorithms. Machine Learning, **39** (2000) 287-308
12. Sutton, R.: Integrated architecture for learning, planning and reacting based on approximating dynamic programming. In Proc. of the 7th ICML (1990) 216-224
13. Sutton, R.S., Barto, A.G.: Reinforcement Learning. An Introduction. The MIT Press (1998)
14. Thrun,S.B.: Efficient Exploration in Reinforcement Learning. Technical Report CMU-CS-92-102. School of Computer Science, Carnegie Mellon University (1992)
15. Vignat C., Bercher, J.-F.: Un estimateur récursif de l'entropie. 17ème Colloque GRETSI, Vannes (1999) 701-704
16. Wiering, M.A., Schmidhuber, J.: Efficient Model-Based Exploration. In Proc. of the 5th International Conference on Simulation of Adaptive Behavior (1998) 223-228
17. Wiatt, J.: Exploration and Inference in Learning from Reinforcement. Ph.D. Thesis. University of Edingburg (1997)