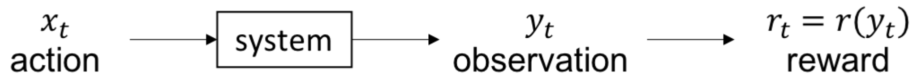
MS&E338 Reinforcement Learning Lecture number - 23 May 2018

Efficient Exploration

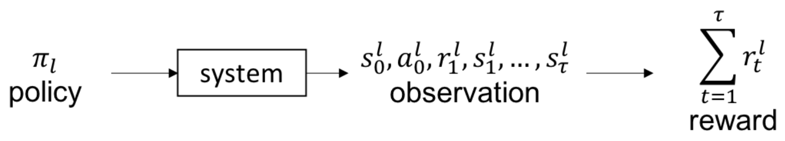
Lecturer: Ben Van Roy *Scribe: Xiaobai Ma, Kunal Menda*

1 Recap: Active Learning and Reinforcement Learning

Active Learning:



Reinforcement Learning:


π_l
policy
system
$s_0^l, a_0^l, r_1^l, s_1^l, \ldots, s_\tau^l$
observation
$\sum_{t=1}^{\tau} r_t^l$
reward

Reinforcement learning is a special case of active learning, where the action x_t is replaced by policy π_l at episode l; the observation y_t is replaced as state, action, reward tuple experienced in this episode; and the reward r_t is replaced by the episodic accumulated reward.

2 UCB/TS for Reinforcement Learning

2.1 Problem Setting

For a finite-horizon MDP defined by $M = (S, A, R, P, \rho, \tau)$, where S is the state space; A is the action space, R is the reward function, P is the transition probability; ρ is the initial state distribution; and τ is the horizon, we make following assumptions:

- Reward is only a function of current state s and current action a: $R \leftarrow R_{s,a}$. In addition, we denote $\bar{R}_{s,a} = E[r_t | s_t = s, a_t = a; R_{s,a}]$
- $r_t \in [0, 1]$

2.2 Regret Bound

With the above assumption, to perform reinforcement learning with uncertainties in P and R, the UCRL method defines a confidence set of possible MDPs as:

$$\Theta = \{M : ||\hat{P}_{s,a} - P_{s,a}||_1 \leq \beta_{s,a} \ \ \& \ \ |\hat{R}_{s,a} - \bar{R}_{s,a}| \leq \beta_{s,a} \forall_{s,a}\}$$

where $\beta_{s,a} = \sqrt{\frac{28|S|\log(2|S||A|L\tau)}{\max(1,N(s,a))}}$ in which L is the episode number and $N(s,a)$ is the number of times that state action pair (s,a) has been experienced. UCRL then optimizes both the policy and the MDP. It can be shown that the expected regret is bounded by:

$$E[Regret(L)] = O(\tau|S|\sqrt{|A|\tau L\log(|S||A|L\tau)})$$

The same regret bound holds for PSRL in which you sample a MDP from a posterior and optimize the action. Then, the episode average regret satisfies:

$$\frac{E[Regret(L)]}{L} \propto \frac{\tau|S|\sqrt{|A|\tau}}{\sqrt{L}}$$

Thus, if we would like to bound the episode average regret by ϵ, the number of episodes we need is given by:

$$L \propto \frac{\tau^3|S|^2|A|}{\epsilon^2}$$

which is a polynomial of τ, $|S|$, $|A|$, and ϵ. This is much better than the ad-hoc exploration methods we discussed before like ϵ-greedy and Botzman exploration, where the number of episodes needed could be exponential in the size of the problem.

Although UCRL and PSRL have the same theoretical regret bound, PSRL usually behaves better empirically. The reason is that UCRL constructs a separate confidence interval for each state-action pair, and the overall confidence set is box-shaped, which can lead to undesirable behavior as discussed in previous lectures. In contrast, PSRL explores more smoothly. Figure 1 shows regret over time in a specific experiment.

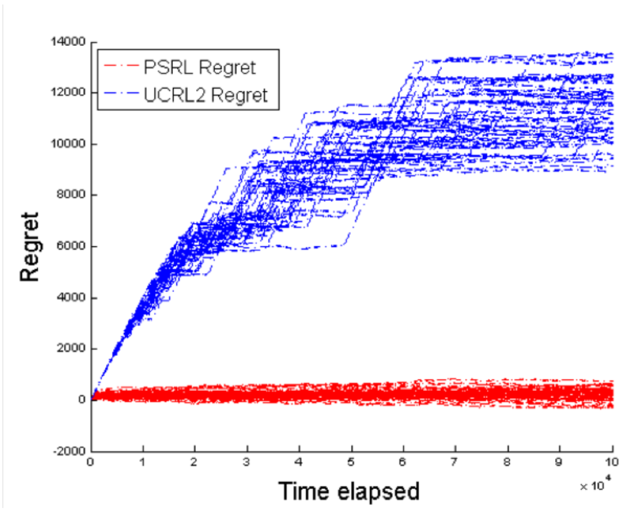


Figure 1: UCRL versus PSRL

3 Efficient Exploration

With knowledge of the polynomial complexity of UCRL, we now segue our discussion to the concept of ‘deep exploration’ strategies. Figure 2 shows the superiority of ‘deep exploration’ strategies, which we consider to be strategies with sample complexity that is polynomial in the size of the problem, with that of common schemes such as ϵ-greedy or Boltzmann exploration. For an intuition about why the naive strategies have such poor performance, we consider the example of a mouse in a maze, shown in Figure 3. A rational mouse is confronted by the decision to either exploit, by going straight to the cheese, or to explore the unknown region that may contain a large positive or negative reward. In either case, it is optimal to head straight to the cheese or to the unknown region. However, an exploration strategy like ϵ-greedy or Boltzmann exploration would waste many time-steps following a random path into the known empty space, neither reaping the benefit of exploration nor exploitation.

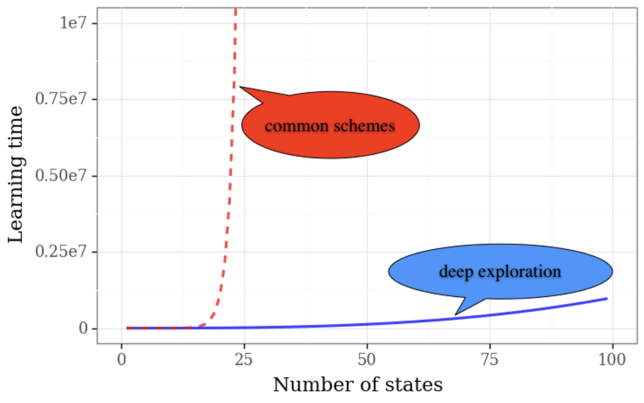


Figure 2: Naive exploration strategies have sample complexity that is exponential in the size of the problem, while ‘deep exploration’ strategies have polynomial sample complexity.

Figure 4 shows a table that enumerates the motivations we may have in some decision problem. If we are interested in immediate reward or learning, it is sensible to use ‘myopic optimization’, that attempts to directly maximize the reward or information gain in the next time-step. If we are interested in rewards that may occur late in the episode, then in deterministic settings we should rely on planning algorithms such as A*. However, if we are interested in learning that may occur late in the episode, we must utilize deep exploration techniques as opposed to naive approaches such as ϵ-greedy exploration, if we are to efficiently scale to large problems.

To see intuitively how PSRL and UCRL achieve ‘deep exploration,’ let us consider the example of the Deep Sea domain, shown in Figure 5. The state space is an N by N grid. In a given cell, the agent can choose the action ‘down1’, which will either transition the agent diagonally down to the left or down to the right, or ‘down2’, which will transition it down and in the opposite horizontal direction as down1, deterministically. The agent does not initially know which action corresponds to which transition. There is a small cost $0.01/N$ incurred each time the down-right action is chosen. There is a large reward (± 1) in the bottom-right corner, but whether it is positive or negative is initially unknown to the agent. In each episode, the agent always starts from the top-left cell, and the episode terminates when the agent reaches the bottom row.

Since this domain is deterministic, if the agent knew that there was a large positive reward in the bottom-right corner, it will always select the right action. If the large reward is negative, the agent will always select the left action.

Consider an ϵ-greedy exploration strategy. the agent will need to choose the appropriate ‘down’ action N times in a row to reach the large reward and uncover whether it is positive or negative. Since the probability of encountering the reward is $(0.5)^N$, the expected number of episodes needed to discover the reward is 2^N. If

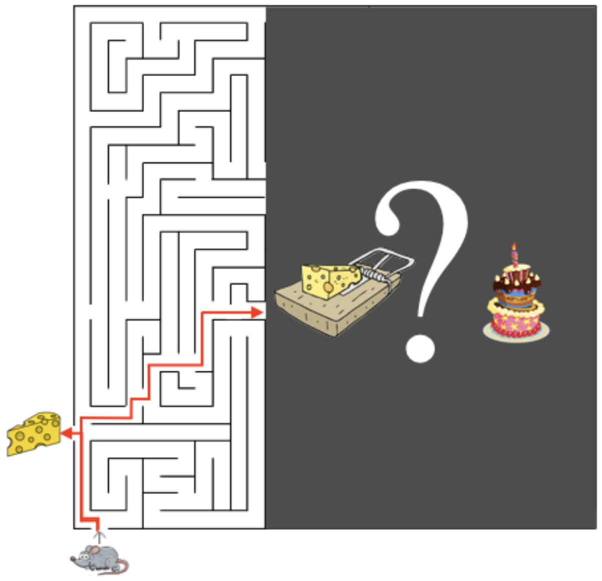
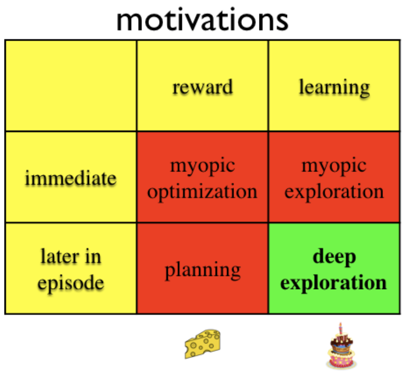


Figure 3: A mouse in a maze would rationally only consider two options: either exploit by going straight to the cheese, or explore by going straight to the unexplored region.



motivations

	reward	learning
immediate	myopic optimization	myopic exploration
later in episode	planning	deep exploration

Figure 4:

ϵ is less than 1, we are biased away from the reward and take even longer to discover the reward. Boltzmann exploration will also have a sample complexity that is exponential in N.

If we use the UCRL algorithm, we assume the best case reward in the bottom right corner. Attempting to reach the goal, the agent will take random actions in the first episode as it does not know which action corresponds to the desired transition in each state. However, once a state has been visited, the agent will never choose the wrong action from this state again. Since it will on average choose correctly in an unknown state 50% of the time, the expected number of episodes needed to discover the sign of the reward is $N/2$.

Similarly, in the case of PSRL, we initially sample a model from our distribution over models, which 50% will prescribe the bottom-right cell as having a large positive reward. Assuming this is the case, we will take $N/2$ episodes to on average to discover the nature of the reward. Since it is the case only 50% of the time, we have an overall expected number of episodes of N.

In this example, we see that ϵ-greedy exploration takes a number of episodes that is exponential in

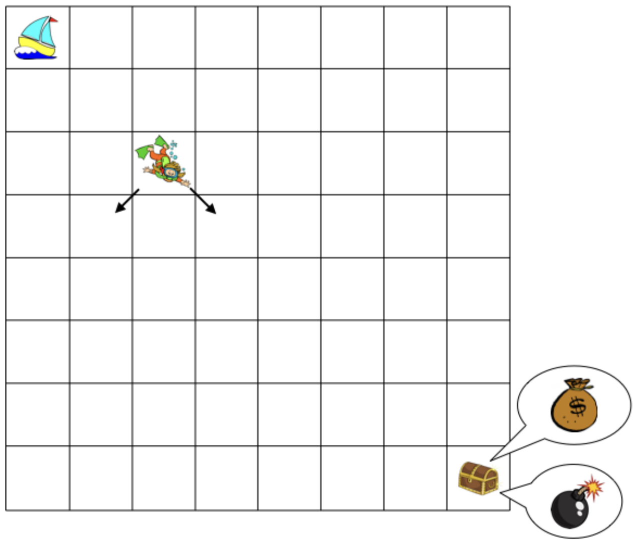


Figure 5: The Deep Sea domain presents demonstrates why UCRL/PSRL dominate naive exploration such as ϵ-greedy in terms of sample complexity.

N to effectively explore the environment, while UCRL/PSRL have linear complexity. Hence, we consider UCRL/PSRL to be 'deep exploration' strategies. It is worth noting that ϵ-greedy need not always have such poor performance, though it is easy to contrive cases in which it does.

4 Roadmap for Future Lectures

So far we have seen clear benefits in opting for PSRL/UCRL, however we have said nothing about tractably implementing these algorithms. UCRL can be something of a computational headache to implement. PSRL, however, is relatively straightforward to implement in the tabular settings, where Beta and Dirichlet distributions represent the distributions over possible models, and updating posteriors is as simple as tracking visitation counts. In future lectures, we will see how PSRL can be applied to problems in which we use function approximators for generalization.