1 Rational RL

Reinforcement learning is one of the pillars of modern artificial intelligence. Artificial intelligence, as a research field, is the quest to build rational machines [CITE: Russell and Norvig, Buckner]. If reinforcement learning is to play a central role in this endeavor, it had better be a method of rational learning. The most successful approach to rational learning is Bayesian inference, and its extension to Bayesian decision theory. The rational authority of Bayesian learning suggests that we should try to understand reinforcement learning from a Bayesian perspective.

In this section, we build on prior work by [CITE: Huttegger] to characterize reinforcement learning from the perspective of Bayesian decision theory.

1.1 Bandits

A k-armed bandit is a formal model of a simple decision problem. The agent interacts with a k-armed slot machine over a period of time. At each time step, the agent may choose to pull any of the levers (arms) on the machine (there needn't be a cost, or buy-in, to doing so). Following lever presses, the machine delivers some reward. The amount of reward depends (perhaps stochastically) on the arm that was pressed. Thus, some arms may, on average, deliver more reward than others. The rewards associated with each arm are entirely independent of one another: in particular, the order in which the levers are pressed has no effect on the reward they deliver.

We consider some fixed time-horizon (corresponding to some fixed number of plays), and set the agent the task of maximizing its cumulative reward (its *return*). How should the agent optimally approach the problem? How would you approach the problem?

The rational course of action depends on what you know. If you know which arm delivers, in expectation, the highest reward, then you could simply pull that arm every time (any other strategy would have lower expected return). If you do not know which arm is best, then a strategy of balanced exploration and exploitation recommends itself: you must both explore the various arms to determine their expected payoffs and exploit the information thus acquired to maximize your expected return. This general strategy suggests two questions: how should you balance exploration and exploitation? And how should you integrate the information you receive about the payoffs as go around pulling levers?

Let us set the former, thornier question aside for the moment and focus on the latter. Given that the bandit's arms are independent, a natural suggestion would be to simply keep a running average of the reward obtained after pulling each arm. Upon pulling arm a at time step t and observing reward r, simply fold r into a running average r_a of the reward obtained from a. This operation can be performed by updating $r_a \leftarrow r_a + \frac{1}{t}r - r_a$. Perhaps unsurprisingly, this simple update rule can be vindicated from a Bayesian perspective. Given the assumption that the rewards are order-independent, ... [Ok, I think I still don't

fully understand the Huttegger stuff]

1.2 Contextual bandits

Bandit problems are particularly simple choice situations. In particular, the best action at a time does not depend on the time, or the environment's state (the environment is, from a formal point of view, stateless). But in most decision problems we face, the environmental state is of crucial importance: the expected utility of taking your umbrella with you is greater in inclement than in mild weather. We are therefore led to generalize the bandit setup so that the distribution of reward associated with an arm may depend on the context (this corresponds to so-called contextual bandit problems).

How should the agent decide what to do in a contextual bandit situation. As Huttegger has shown, there is not much for the agent to do if it cannot track the environmental state. An agent with no access to the state cannot in general converge on the optimal choice behavior in a contextual bandit. This is to be expected: intuitively, the optimal action at any given point depends entirely on the state of the environment; it is hard to see how an agent blind to this state could nonetheless select the right action with high probability.

If we give the agent access to the current state, however, then it can condition its learning on the state. By keeping track of the average reward obtained by choosing action *a in state s*, the agent can once again learn to behave optimally (see [CITE: Huttegger, 71]).

1.3 Markov Decision Processes

This decision problem remains rather impoverished, however. In real-world scenarios, one's choices impact the state of the environment. The dependence of environmental state on one's action drastically increases the complexity of the learning problem, by making later choice situations depends on earlier choices. Bandit problems—in which future states do not depend on what the agent chooses—are amenable to so-called *greedy* strategies: even relative to the long term goal of maximizing expected return (cumulative reward), the optimal action is that which maximizes expected *immediate* reward. This is of course not the case in many real-world problems, where choosing actions that maximize immediate reward can have disastrous long-term consequences. How, then, should we model decision problems in which the agent can influence the state of the world, and in particular, the next choice situation?

The next step on the ladder of sequential decision making (at which we leave Hutteger's work behind) is that of Markov Decision Processes (MDPs). In an MDP, the environment can be in one of several given states and the agent has in each state a number of actions available. The agent's actions are associated, in each state, with a reward (or a distribution over possible rewards) and with a (distribution over) next states of the environment. In this way we model the agent's influence upon its environment, and the environment's feedback on the agent's actions. MDPs also make the titular Markov assumption: the reward

and next state attached to an action depend solely on the present state. In particular, the best action to take in each state does not depend on the history of states leading up to the present. This assumption may appear unduly restrictive. While the Markov assumption is undoubtedly substantial, MDPs can nonetheless model a wide range of phenomena arising in cognitive science, economics, robotics, and optimal control. Moreover, the central class of algorithms for handling MPDs—reinforcement learning algorithms—has been extended to cover deviations from the Markov assumption. In any case, the dependence of future states on present choices is distinctive of MDPs, and is what makes the decision problem both more flexible and more difficult than bandit problems. In particular, reward-greedy strategies are no longer appropriate: it may be in the agent's interest to sacrifice short-term gains to achieve higher overall returns.

Unlike in the (contextual) bandit cases, there is no obviously rationally optimal solution to MPDs. By far the most popular class of solutions to the MPD problem come from the field of reinforcement learning. Reinforcement learning develops and studies methods for solving MDP problems by learning from experience. Model-free algorithms maintain estimates of the value of different actions or states, and use these estimates to decide what actions to take. Value estimates represent the agent's "best guess" concerning the return that follows an action. Maintaining value estimates reduces the MDP problem to one where greedy behavior is appropriate (modulo exploration): instead of recording the average reward attaching to an action (in a state), value functions track the richer (and harder to compute) expected return of an action. Thus, model-free algorithms buy simplicity in decision-making (greedy strategies) at the cost of more complex learning algorithms.

Model-based algorithms maintain a representation of their environment and of the effects of their actions on it. When deciding what to do, they use the model to compute and execute the best course of action. The model is learned from interacting with the environment, in similar ways to how value functions are learned.