Parallel Reinforcement Learning

R. Matthew Kretchmar
Mathematics and Computer Science, Denison University
Granville, OH 43023, USA

Abstract

We examine the dynamics of multiple reinforcement learning agents who are interacting with and learning from the same environment in parallel. Due to the stochasticity of the environment, each agent will have a different learning experience though they should all ultimately converge upon the same value function. The agents can accelerate the learning process by sharing information at periodic points during the learning process.

Keywords: Reinforcement Learning, Parallel Agents, Multi-Agent Learning

1 Introduction

Here we investigate the problem of multiple reinforcement learning agents attempting to learn the value function of a particular task in parallel. Each agent is simultaneously engaging in a separate learning experience on the same task. It seems intuitive that each agent's learning experience can be accelerated if the agents share information with each other during the learning process. We examine the complexities of this information exchange and propose a simple algorithm that successfully demonstrates accelerated learning performance among parallel reinforcement learning agents.

In the remainder of the Introduction, we briefly review the problem of reinforcement learning and discuss previous efforts in parallel reinforcement learning. Section 2 presents the parallel reinforcement learning problem in the context of the n-armed bandit task. Section 3 provides an algorithmic solution to parallel reinforcement learning. In Section 4, we present empirical evidence of accelerated learning on the n-armed bandit task. Finally, Section 5 suggests possible avenues of future research.

Reinforcement learning (RL) is the process of learning to behave optimally via trial-and-error experience. An agent interacts with an environment by observing states, s, and selecting actions, a where the action choice moves the agent to new states in the environment. The agent also receives a reward r per each state-action choice. The goal of the agent is to maximize the *sum* of all rewards experienced. The major challenge in reinforcement learning is to have the agent not only defer immediately large rewards for larger future rewards, but to also choose actions that lead to the states with the opportunity for larger future rewards. The interested reader is referred to [9] for a comprehensive introduction to reinforcement learning.

Despite its apparent simplicity, there has been surprisingly little work in parallel reinforcement learning. Most of the research concerns multiple agents learning *different* but inter-related tasks. Littman studies competing RL agents within the context of Markov games [4, 5]. Sallans and Hinton [8] study agents who cooperate to solve different parts of a larger task. Claus and Boutilier [3] and later Mundhe and Sen [6] also examine the various complex inter-relations of multiple agents in cooperating to solve a common task. The common feature of all this existing work is that the agents are solving different parts of a task or are working in an environment that is altered by the actions of other agents; in this work we concentrate on a simplified version of the problem in which multiple agents independently interact with a stationary environment. Only in Bagnell [1], do we see some initial work along this line; here multiple RL robots learn in parallel by broadcasting learning tuples in real time. However in Bagnell's work parallel RL is only used as a means to study other behavior; parallel RL is not the object of investigation.

2 The Parallel Reinforcement Learning Problem

We introduce the problem of parallel reinforcement learning using the n-armed bandit task to illustrate the concepts. The n-armed bandit task, named for slot machines, has been studied extensively in the fields of mathematics, optimization, and machine learning [2, 7, 9]. We follow the experiments of Sutton and Barto [9] in constructing simple agents that use *action-value* methods to estimate the payoff(reward) of each arm(action).

2.1 Reinforcement Learning and the n-armed Bandit

On each trial, the agent selects one arm (action a) from a set of n arms and receives a payoff r as a result of that action; the payoff is a normally distributed random variable r with mean $Q^*(a)$ and standard deviation 1. The agent maintains an estimate of the mean payoff of bandit arm a by averaging the rewards received by pulling arm a:

$$Q_t(a) = \frac{r_1 + r_2 + ... + r_{k_a}}{k_a} \quad (1)$$

$$t = \sum_{i=0}^{n-1} k_i \quad (2)$$

where t is the number of total trials counting all actions, k_a is the number of these trials allocated specifically to action a, and $r_1, r_2, ..., r_{k_a}$ are the individual samples or rewards experienced when choosing action a over the k_a different trials. In order to avoid storing all k_a rewards for each of the n arms, we can use an incremental approach that stores only the current estimate, $Q_t(a)$, and the number of trials for each arm, k_a. The on-line, incremental update rule is then:

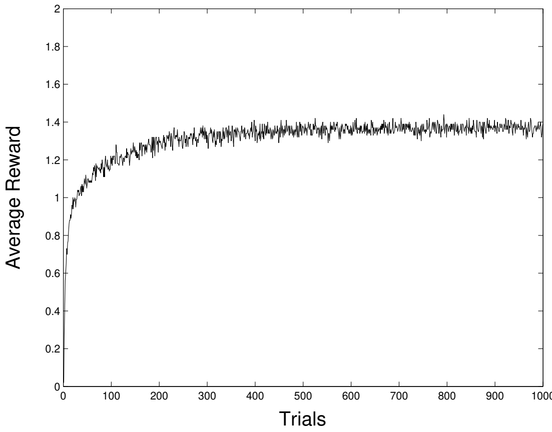
$$Q_{t+1}(a) = \begin{cases} \frac{r_{k_a+1} + k_a Q_t(a)}{k_a+1} & \text{if action } a \text{ is selected} \\ Q_t(a) & \text{otherwise.} \end{cases} \quad (3)$$

Figure 1 shows the learning performance of a single RL agent interacting with a 10-armed bandit. We use an $\epsilon - greedy$ policy ($\epsilon = 0.1$) to average 2000 different experiments where each contains 1000 trials. For each experiment, $n = 10$ bandits are created randomly with Q^* sampled from $\mathcal{N}(1.0, 1.0)$, a normal distribution with mean 1.0 and standard deviation 1.0.

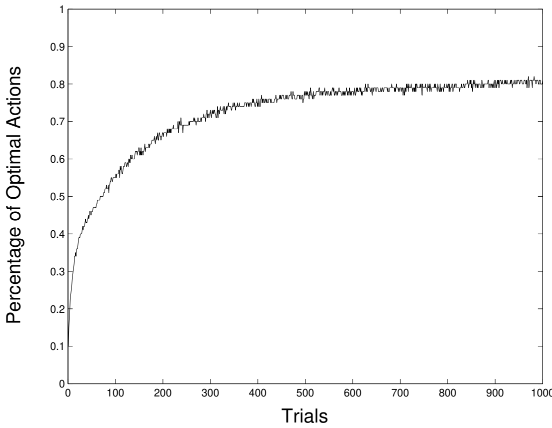
It is clear that the value of an agent's payoff estimate for a particular action, $Q(a)$, is directly related to the number of trials allocated to this action, k_a. As the agent gains more experience, its estimate of the reward for each arm, $Q(a)$, approaches the true mean, $Q^*(a)$.

2.2 The Problem of Parallel Learning

The experiment of the previous section reveals the importance of the agent's *experience*. The number of trials is the currency by which an agent can gauge it's success; the more trials, the better the reward estimates and hence the more probable the agent is able to select the optimal action. Clearly, any change to the basic algorithm that provides the agent with more experience can improve the agent's learning performance.



(a) Average Reward



(b) Percentage of Optimal Actions

Figure 1: Single Agent in 10-armed Bandit Task

We now consider the case where multiple agents are learning the same n-armed bandit task in parallel. Keep in mind that the agents are not experiencing the *exact same* series of payoffs; each agent is sampling independently and also able to allocate its t total samples over the n actions differently. Thus each agent is accumulating a different experience.

For illustration, we consider the case of two agents, `Agent0` and `Agent1`, and a 1-armed bandit (one action) with payoff $Q^*(0) = 1$. At some point during the learning, the state of the two agents is as follows:

- `Agent0` has selected action 0 twice and received payoffs of 1.1 and 1.05. `Agent0` estimates the payoff to be $Q(0) = \frac{1.1+1.05}{2} = 1.075$.

- `Agent1` has selected action 0 once and received a payoff of 0.9. `Agent1` estimates the payoff to be $Q(0) = \frac{0.9}{1} = 0.9$.

We can say that `Agent0`'s estimate is probably more accurate than is `Agent1`'s because `Agent0` has twice as much learning experience with action 0. Since each agent's trials were independent, we can also claim that, between the two agents, there are three trials (samples). The agents could then combine their experience as follows:

$$\begin{aligned}
\text{Total Experience} &= \texttt{Agent0}\text{'s experience} \\
&\quad +\texttt{Agent1}\text{'s experience.} \\
&= k_0(\texttt{Agent0}) + k_0(\texttt{Agent1}) \\
&= 2 + 1 = 3 \\
\text{Combined Estimate} &= \texttt{Agent0}\text{'s estimate} \\
&\quad \text{weighted by its experience} \\
&+ \texttt{Agent1}\text{'s estimate} \\
&\quad \text{weighted by its experience.} \\
&= 1.075\frac{2.0}{3.0} + 0.9\frac{1.0}{3.0} \\
&= 1.017
\end{aligned}$$

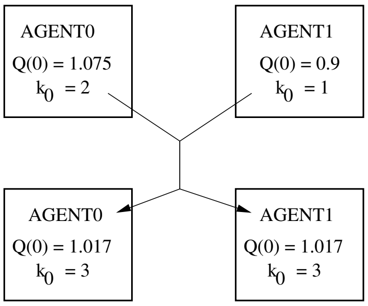


Figure 2: Two Agents Combining Experience

We depict this exchange of information in Figure 2. However, this notion is not entirely correct; a problem arises when we attempt to further combine shared experience. Neither `Agent0` nor `Agent1` truly have three trials of learning experience. It is true that they have a combined three trials of experience upon which to base their estimates, but this is distinct from the case in which each agent has three *separate* trials of experience. A problem will arise because now the agents' experience is not independent.

This subtle problem is elucidated when we consider that these same two agents meet again and decide to share learning experience in the same way; each agent comes away from the second swapping episode believing that it now has six trials of experience upon which to base an estimate. These agents could continue to swap information indefinitely and to reach an "infinite amount" of experience when, in fact, there are still only the original three trials from which it is all based. If one of these two agents were to swap information with a third agent that has 100 actual trials of experience to it's credit, the third agent's information would be statistically overwhelmed by the correspondingly larger accumulated experience of the first agent – even though this first agent really only possesses three actual trials of experience.

3 The Parallel Reinforcement Learning Solution

To overcome this problem, we must have each agent keep track of two sets of parameters: one set for the actual independently experienced trials of that particular agent, and an additional set for combined trials among all other agents [1]. A better way to depict the agents is shown in Figure 3. Each agent now maintains $\tilde{Q}(a)$ and $\tilde{k}_a$ per action to keep track of *only* those trials directly experienced by this agent. Added now are $\hat{Q}(a)$ and $\hat{k}_a$ which are the combined estimates of *all other* agents' experience and parameters. Specifically, $\hat{k}_a$ is the total number of trials for action a experienced by all other agents, and $\hat{Q}(a)$ is the average payoff estimate for all other agents.

This new arrangement enables several important computations that were not possible before:

1. The agents can accurately share accumulated experience by keeping separate parameters for their own independent experience (trials) and the combined experience of all other agents.
2. The agents can compute an accurate estimate based upon the global experience. This estimate can be computed from a weighted average of the agent's own independent experience and the accumulated experience of all other agents:

$$Q_t(a) = \frac{\tilde{Q}_t(a) * \tilde{k}_a + \hat{Q}_t(a) * \hat{k}_a}{\tilde{k}_a + \hat{k}_a}$$

[1] We choose not to include the agent's own experience in its combined experience results. This way, the agent can continue to learn with additional trials and still effectively remember and combine the experience of other agents.

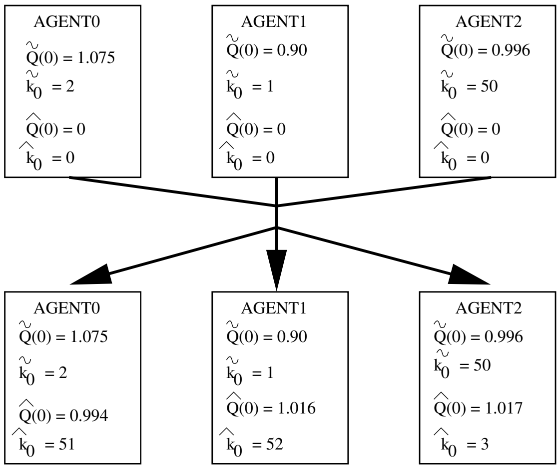


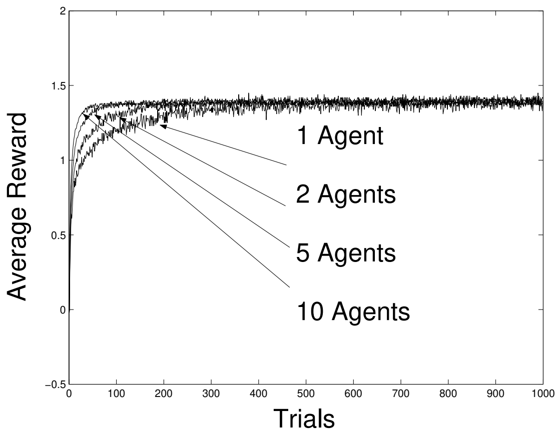
Figure 3: Storing Independent Experience Separately from Shared Experience

3. The agents can continue to accurately gain new experience by adding to $\tilde{Q}_t(a)$ and $\tilde{k}_a$ and thereby continue to improve their estimates of $Q_t(a)$ and k_a even though they may not be able to continue to share parameters with other agents.

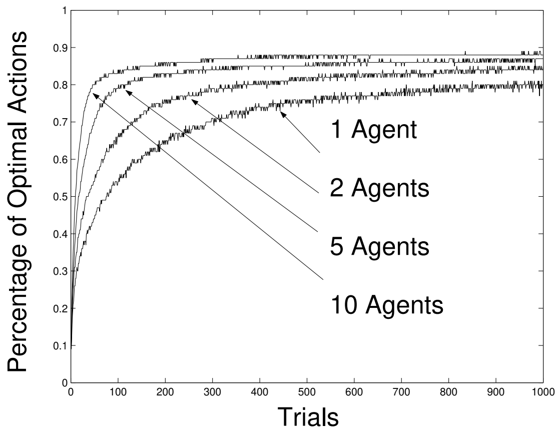
4 n-armed Bandit Results

In this section we empirically demonstrate the improvement of allowing parallel agents to share learned experience in the n-armed bandit task. As before, each agent experiences 1000 trials (actions) in each of 2000 different experiments (the results are averaged over the 2000 experiments). For each experiment, we randomly select ten ($n = 10$) bandits with average payoffs ($Q^*(a)$) chosen from $\mathcal{N}(1.0, 1.0)$. In this case we vary the number of agents from 1, 2, 5, and 10. The agents share accumulated experience after every trial; thus there are 1000 separate episodes of parameter sharing among all the agents – one after each of the 1000 trials.

Figure 4 shows the average payoff and percentage of optimal actions of all the agents during the experiments. Clearly, the individual agent performs the worst as it can only use its own experience. As expected, adding more agents accelerates the learning process because there is a larger pool of accumulated experience upon which to base future estimates. The experiment with 10 parallel agents learns the fastest.



(a) Average Reward



(b) Percentage of Optimal Actions

Figure 4: Parallel Agents in 10-armed Bandit Task

5 Directions of Future Work

While the concept of parallel reinforcement learning is relatively simple and its benefits are obvious, there has been almost no work in this area. There are numerous opportunities for extended work; here are some currently under investigation:

- Quantify the possible theoretical *speed-up* with parallel agents.
- Investigate the increased complexity between exploitation and exploration. With parallel agents sharing information, there is additional pressure for more agents to exploit the same actions instead of diversely exploring.
- Extend the process to multi-state tasks. We expect an even greater benefit for episodic tasks of more than one state.

- There seems to be a curious inversion effect where the performance of the group as a whole increases if the agents share information less frequently. We hypothesize dynamics similar to the "island models" of genetic algorithms that prevent the system as a whole from prematurely converging upon a non-optimal solution.

References

[1] J. Andrew Bagnell. A robust architecture for multiple-agent reinforcement learning. Master's thesis, University of Florida, 1998.

[2] D. A. Berry and B. Fristedt. *Bandit Problems*. Chapman and Hall, 1985.

[3] Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*. AAAI, 1998.

[4] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 157–163, 1994.

[5] Michael L. Littman. Value-function reinforcement learning in markov games. *Journal of Cognitive Systems Research*, 2001.

[6] Manisha Mundhe and Sandip Sen. Evaluating concurrent reinforcement learners. In *Proceedings of the Fourth International Conference on Multiagent Systems*, pages 421–422. IEEE Press, 2000.

[7] K. S. Narendra and M. A. L. Thathachar. *Learing Automata: An Introduction*. Prentice-Hall, 1989.

[8] Brian Sallans and Geoffrey Hinton. Using free energies to represent q-values in a multiagent reinforcement learning task. In *Advances in Neural Information Processing Systems 13 (NIPS 2001)*, volume 13. MIT Press, 2001.

[9] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.