Chapter 15
Exploitation-Oriented Learning XoL: A New Approach to Machine Learning Based on Trial-and-Error Searches

Kazuteru Miyazaki
National Institution for Academic Degrees and University Evaluation, Japan

ABSTRACT
Exploitation-oriented Learning XoL is a new framework of reinforcement learning. XoL aims to learn a rational policy whose expected reward per an action is larger than zero, and does not require a sophisticated design of the value of a reward signal. In this chapter, as examples of learning systems that belongs in XoL, we introduce the rationality theorem of profit Sharing (PS), the rationality theorem of reward sharing in multi-agent PS, and PS-r*. XoL has several features. (1) Though traditional RL systems require appropriate reward and penalty values, XoL only requires an order of importance among them. (2) XoL can learn more quickly since it traces successful experiences very strongly. (3) XoL may be unsuitable for pursuing an optimal policy. The optimal policy can be acquired by the multi-start method that needs to reset all memories to get a better policy. (4) XoL is effective on the classes beyond MDPs, since it is a Bellman-free method that does not depend on DP. We show several numerical examples to confirm these features.

INTRODUCTION
The approach, called reinforcement learning (RL), is much more focused on goal-directed learning from interaction than are other approaches to machine learning (Sutton and Barto, 1998). It is very attractive since it can use Dynamic Programming (DP) to analyze its behavior in the Markov Decision Processes (MDPs) (Sutton, 1988; Watkins and Dayan, 1992; Ng et al., 1999; Gosavi, 2004; Abbeel and Ng, 2005). We call these methods that are based on DP DP-based RL methods. In general, RL uses a reward as a teacher signal for its learning. The DP-based RL method aims to optimize its behavior under the values of reward signals that are designed in advance.

We want to apply RL to many real world problems more easily. Though we know some
important applications (Merrick et al., 2007), generally speaking, it is difficult to design RL systems to fit on a real world problem. We think that the following two reasons concern with it. In the first, the interaction will require many trial-and-error searches. In the second, there is no guideline how to design the values of reward signals. Though they are not treated as important issues on theoretical researches, they are able to be a serious issue in a real world application. Especially, if we have assigned inappropriate values to reward signals, we will receive an unexpected result (Miyazaki and Kobayashi, 2000). We know the Inverse Reinforcement Learning (IRL) (Ng and Russell, 2000; Abbeel and Ng, 2005) as a method related to the design problem of the values of reward signals. If we input our expected policy to the IRL system, it can output a reward function that can realize the policy. IRL has several theoretical results, i.e. apprenticeship learning (Abbeel and Ng, 2005) and policy invariance (Ng et al., 1999).

On the other hand, we are interested in the approach where reward signals are treated independently and do not require a sophisticated design of the values of them. Furthermore, we aim to reduce the number of trial-and-error searches through strongly enhancing successful experiences. We call it Exploitation-oriented Learning (XoL). As examples of learning systems that belong in XoL at the environments, we introduce the rationality theorem of PS, the rationality theorem of PS in multi-agent environments, and PS-r*. We show several numerical examples to support how to use these methods.

**PROBLEM FORMULATIONS**

**Notations**

Consider an agent in some unknown environment. The agent senses a set of discrete attribute-value pairs and performs an action in some discrete varieties. The environment provides a reward signal to the agent as a result of some sequence of an action. We denote the sensory inputs as $x, y, \ldots$ and actions as $a, b, \ldots$. A sensory input and an action constitute a pair that is termed as a rule. We denote the rule “if $x$ then $a$” as $xa$. In Profit Sharing (PS), a scalar weight, that indicates the importance of the rule, is assigned to each rule. The weight of the rule $xa$ is denoted as $\omega_{xa}$. The function that maps sensory inputs to actions is termed a policy. We call a policy rational if and only if the expected reward per an action is larger than zero. Furthermore, a useful rational policy is a rational policy that is not inferior to the random walk (RA) where the agent selects an action based on the same probability to every sensory input. The policy that