SICE Annual Conference 2008
August 20-22, 2008, The University Electro-Communications, Japan

How to recommend preferable solutions of a user in interactive reinforcement learning ?

Tomohiro Yamaguchi [1] and Takuma Nishimura [2]

[1 2] Nara National College of Technology, Nara, Japan
(Tel : +81-743-55-6140; E-mail: [1]yamaguch@info.nara-k.ac.jp, [2] nishimura@info.nara-k.ac.jp)

Abstract: We propose a new method of recommending preferable solutions of a user in interactive reinforcement learning. Interactive reinforcement learning is different from normal reinforcement learning in that a human gives the reward function to the learner interactively. It is that the reward function may not be fixed for the learner if an end-user changes his mind or his preference. However, most of previous reinforcement learning methods assume that the reward function is fixed and the optimal solution is unique, so they will be useless in interactive reinforcement learning with such an end-user. To solve this, it is necessary for the learner to estimate the user's preference and to consider its changes. This paper proposes a new method how to match an end-user's preference solution with the learner's recommended solution. Experiments are performed with twenty subjects to evaluate the effectiveness of our method. As the experimental results, a large number of subjects prefer each e*very-visit-optimal* solution than the optimal solution. On the other hand, a small number of subjects prefer each e*very-visit-non-optimal* solution. We will discuss the reason why the end-users' preferences are divided into two groups.

Keywords: Reinforcement Learning, Interactive, every-visit-optimality, coarse to fine, recommendation.

1. INTRODUCTION

In field of robot learning [1], interactive reinforcement learning method in that reward function denoting goal is given interactively has worked to establish the communication between a human and the pet robot AIBO. The main feature of this method is the interactive reward function setup which was fixed and build-in function in the main feature of previous reinforcement learning methods. So the user can sophisticate reinforcement learner’s behavior sequences incrementally.

Shaping [2-4] is the theoretical framework of such interactive reinforcement learning methods. Shaping is to accelerate the learning of complex behavior sequences. It guides learning to the main goal by adding shaping reward functions as sub goals. Previous shaping methods [3,4] have three assumptions on reward functions as following;

1. Main goal is given or known for the designer.
2. Sub goals are assumed as shaping rewards those are generated by potential function to the main goal [3].
3. Shaping rewards are policy invariant (not affecting the optimal policy of the main goal) [4].

However, these assumptions will not be true on interactive reinforcement learning with an end-user. Main reason is that it is not easy to keep these assumptions while the end-user gives rewards for the reinforcement learning agent. It is that the reward function may not be fixed for the learner if an end-user changes his mind or his preference. However, most of previous reinforcement learning methods assume that the reward function is fixed and the optimal solution is unique, so they will be useless in interactive reinforcement learning with an end-user.

To solve this, it is necessary for the learner to estimate the user's preference and to consider its changes. This paper proposes a new method how to match an end-user's preference solution with the learner's recommended solution. Our method consists of three ideas. First, we assume *every-visit-optimality* as the optimality criterion of preference for most of end-users. Second, to cover the end-user's preference changes after the reward function is given by the end-user, the learner prepares *various policies* [6] by generating variations of the reward function under *every-visit-optimality*. Third, we propose *coarse to fine recommendation* strategy for guiding the end-user's current preference.

To examine these ideas, we perform the experiment with twenty subjects to evaluate the effectiveness of our method. As the experimental results, a large number of subjects prefer each *every-visit-optimal* solution than the optimal solution. On the other hand, a small number of subjects prefer each *every-visit-non-optimal* solution. We discuss the reason why the end-users' preferences are divided into two groups.

2. MODEL BASED REINFORCEMENT LEARNING

This chapter describes the framework of reinforcement learning in our research. Fig. 1 shows an overview of our model-based reinforcement learning system. Note that s is an observed state, a is an executed action, and Rw is an acquired reward. In Fig. 1, our learning agent consists of three blocks which are model identification block, optimality of policies block and policy search block. The details of these blocks are described in following section. The novelty of our method lies in optimality of policies which is described in the section 2.2 and the method of searching policies described in the section 2.3.

2.1 Model identification

In model identification block, the state transition probabilities $P(s' \mid s,a)$ and reward function $R(s,a)$ are estimated incrementally by observing a sequence of *(s,a,r)*. This estimated model is generally assumed Markov Decision Processes (MDP) [5]. MDP model is defined by of following four elements.

1. Set of states : $S = \{s_0, s_1, s_2, \cdots, s_n\}$
2. Set of actions : $A = \{a_0, a_1, a_2, \cdots, a_m\}$
3. State transition probabilities : $P(s' \mid s, a)$ probability of occurring state s' when execute action a at state s
4. Reward function : $R(s,a)$ acquired reward when execute action a at state s

2.2 Optimality of policies

Optimality of policies block defines the optimality of the learning policy. In this research, a policy which maximizes average reward is defined as an optimal policy. Eq. (1) shows the definition of average reward.

$$g^{\pi}(s) \equiv \lim_{N\to\infty} E\left(\frac{1}{N}\sum_{t=0}^{N-1} r_t^{\pi}(s)\right) \qquad (1)$$

where N is the number of step , $r_{\tau}^{\pi}(s)$ is the expected value of reward that an agent acquired at step t where policy is π and initial state is s and $E(\ )$ denotes the expected value.

Then we introduce *every-visit-optimal* (*ev-optimal*) policy based on average reward. The detail of *ev-optimal* is described in section 3.1.

2.3 Policy search

Policy search block searches *ev-optimal* policies on an identified mode according to optimality of policies. The detail of this block is described in section 3.2.

3. PREPARING VARIOUS POLICIES

This chapter describes the definition of *various policies* and the method for searching *various policies*.

3.1 Definition of every-visit-optimal

First, we define *every-visit-optimal (ev-optimal)* for the new policy learning criterion. *Ev-optimal* policy is the optimal policy that visits every reward in the reward function. For example, if the reward function has three rewards, the *ev-optimal* policy is the largest average reward one which visits these three rewards.

3.2 Definition of various policies by ev-optimal

Various policies are defined by following two steps.

(1) Enumerate the all subsets of the reward function.
(2) Search an *ev-optimal* policy for each subset of the reward function.
(3) Collect all *ev-optimal* policies.

Fig. 2 illustrates the process for searching *various policies*. When a reward function is identified as {Rw1, Rw2}, enumerated subsets of the function are {Rw1}, {Rw2}, {Rw1, Rw2} in step 1. Then an ev-optimal policy is decided for each subset of the reward function in step 2. At last, these ev-optimal policies are collected as *various policies*. The number of policies in the *various policies* is $2^r - 1$ where r is the number of rewards in the model.

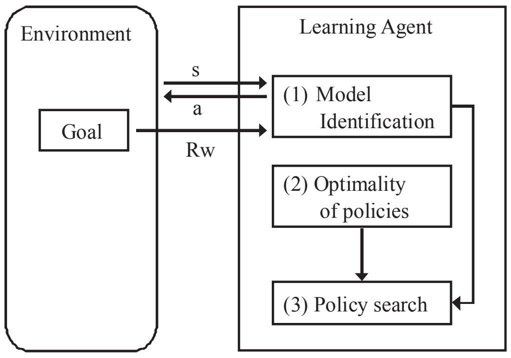


Fig. 1 Framework of reinforcement learning

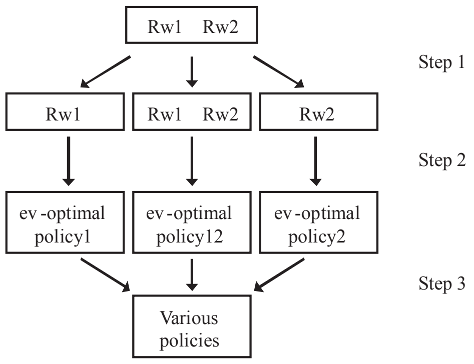


Fig. 2 Process for searching various policies

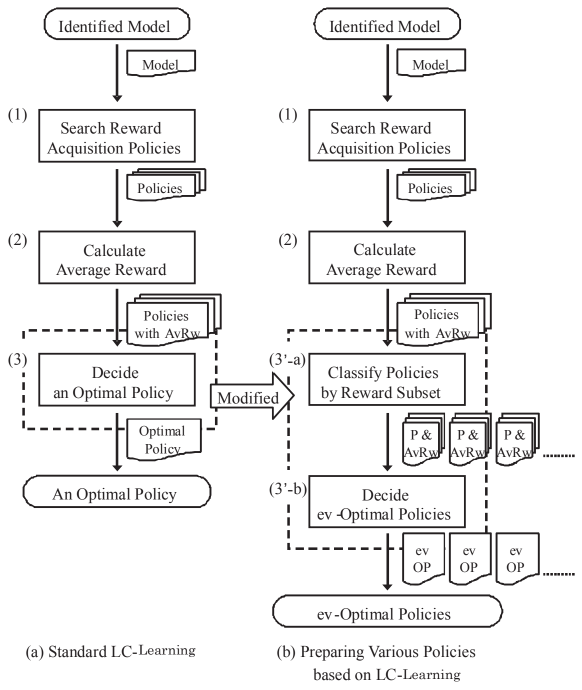


Fig. 3 Algorithm for preparing various policies

3.3 Searching various policies

This section describes our various policies search method based on LC-Learning[6,7]. LC-Learning is one of the average reward model-based reinforcement learning methods. The features of LC-Learning are following; (1) Breadth search of an optimal policy started by each reward rule. (2) Calculating average reward by reward acquisition cycle of each policy.

Fig. 3 (a) shows standard LC-Learning algorithm. Previous LC-Learning decides an optimal-policy by following three steps.

(1) Search policies that have reward acquisition cycle.
(2) Calculate average reward of searched policies
(3) Decide an optimal policy that has maximum average reward.

Fig. 3 (b) shows algorithm for preparing *various policies* based on LC-Learning. Major difference from standard LC-Learning is the third step. Next, we describe these three steps.

(1) Search reward acquisition policies

In this step, reward acquisition policies are searched by converting a MDP into the tree structures where reward acquisition rules are root rule. Fig. 4 to Fig. 6 shows example. The MDP shown in Fig. 4 which consists of four states, six rules and two rewards is converted into the two tree structures shown in Fig. 5. In a tree structure, the path from the root node to the state that is same state to the initial node is the policy. In stochastic environment, some of the rule is deliquesce stochastically. In such case, path from parent node of stochastic rule to the state that is already extracted is part of a policy that contains the stochastic rule. Fig. 6 shows all reward acquisition policies in Fig. 4.

(2) Calculate average reward

In this step, average reward of each policy is calculated by using occurring probability of each state of the policy. Occurring probability of a state is expected value of the number of transiting the state during the agent transit from the initial state to the initial state. Eq. (2) shows definition of the occurring probability of state s_j where initial state is s_i. Occurring probabilities of each state is calculated approximately by value iteration using eq. (2).

$$P_o(s_j, s_i) = \begin{cases} 1 & (j = i) \\ \sum_{s_k} P_o(s_k, s_i) P(s_j \mid s_k, a_k) & (j \neq i) \end{cases} \quad (2)$$

Where a_k is the action that is executed at state s_k.

$$g^{\pi}(s_i) = \frac{\sum_{s_j} P_o(s_j, s_i) R(s_j, a_j)}{\sum_{s_j} P_o(s_j, s_i)} \quad (3)$$

The average reward of policies is calculated by eq. (3) using occurring probability calculated by eq. (2).

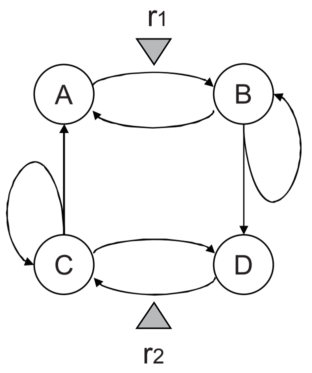


Fig. 4 An example of MDP model

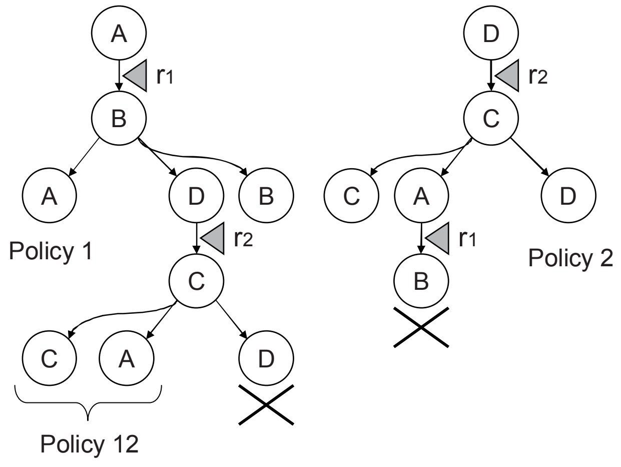


Fig. 5 Searching reward acquiring policies

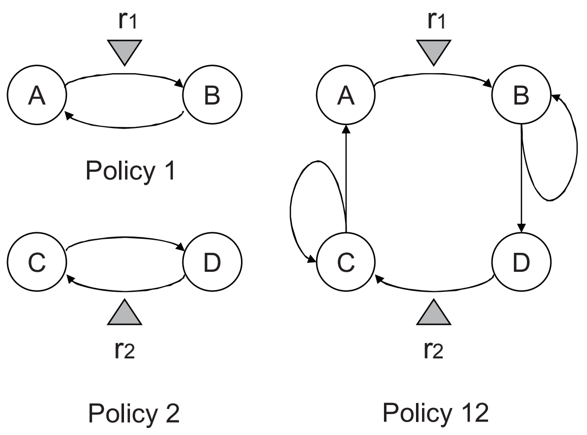


Fig. 6 Three kind of reward acquiring policies

(3'-1) Classify policies by reward subset

In this step, all policies searched by step 1 are classified by acquisition reward set.

(3'-2) Decide ev-optimal policies

In this step, an *ev-optimal* policy is decided for each group classified in step (3'-1). Each *ev-optimal* policy is a policy that had maximum average reward in the each group.

3.3 Searching various policies

This section summarizes our various policies search method based on LC-Learning [6,7]. LC-Learning is one of the average reward model-based reinforcement learning methods. The features of LC-Learning are following; (1) Breadth search of an optimal policy started by each reward rule. (2) Calculating average reward using reward acquisition cycle of each policy.

4. PLAN RECOMMENDATION

This chapter describes the plan recommendation system and the *coarse to fine recommendation* strategy. In this chapter, a *goal* is a reward to be acquired, and a *plan* means a cycle that acquires at least one reward in a policy.

4.1 Overview of the plan recommendation system

Fig. 7 shows an overview of the plan recommendation system. When a user input several goals to visit constantly, they are converted to the set of rewards in the round-trip plan task block for the input of interactive LC-learning block. After various policies are prepared, each policy is output as a plan to the user.

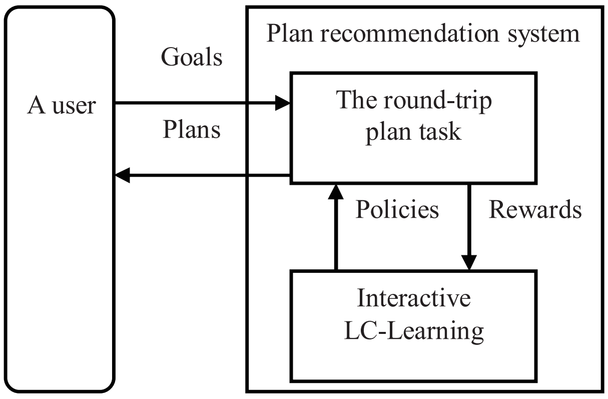


Fig. 7 Overview of the plan recommendation system

4.2 Grouping various plans by the visited goals

After classified policies by reward subset in section 3.2, they are merged into group by the number of acquired reward. Fig. 8 shows grouping various policies by the number of visited reward. When three goals are input by a user, they are converted into three kinds of reward as Rw1, Rw2, and Rw3. Then, Group1 in Fig. 8 holds various policies acquiring only one reward among Rw1, Rw2, or Rw3. Group2 holds various policies acquiring two kinds of reward among Rw1, Rw2, or Rw3, and Group3 holds various policies acquiring Rw1, Rw2, and Rw3.

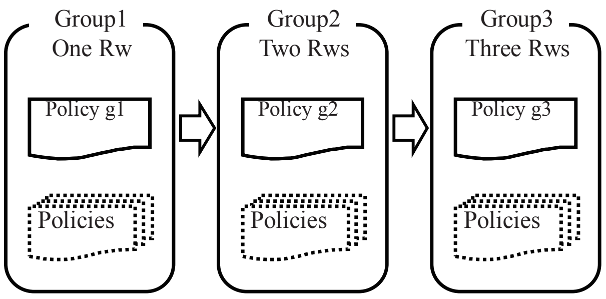


Fig. 8 Grouping various policies

4.3 Coarse to fine recommendation strategy

After grouping various plans by the number of visited goals, they are presented to the user sequentially for selecting the most preferable plan. We call the way to decide this order as *recommendation strategy*. In this paper, we propose *coarse to fine* recommendation strategy that consists of two steps, coarse recommendation step and fine recommendation step.

(1) Coarse recommendation step

For the user, the aim of this step is to select a preferable group. To support the user's decision, the system recommends a representative plan in each selected group to the user.

Fig. 8 shows a coarse recommendation sequence when a user changes his preferable group as Group1, Group2, and Group3 sequentially. When the user selects a group, the system presents a representative plan in the group as recommended plan.

(2) Fine recommendation step

For the user, the aim of this step is to decide the most preferable plan in the selected group in previous step. To support the user's decision, the system recommends plans among his selected group to the user. Fig. 9 shows a fine recommendation sequence when a user select his preferable group as Group2.

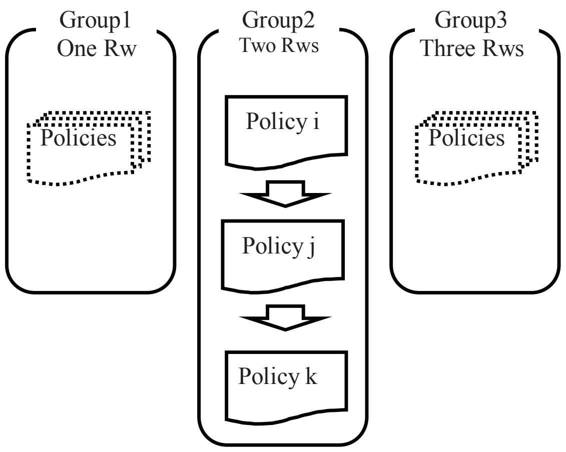


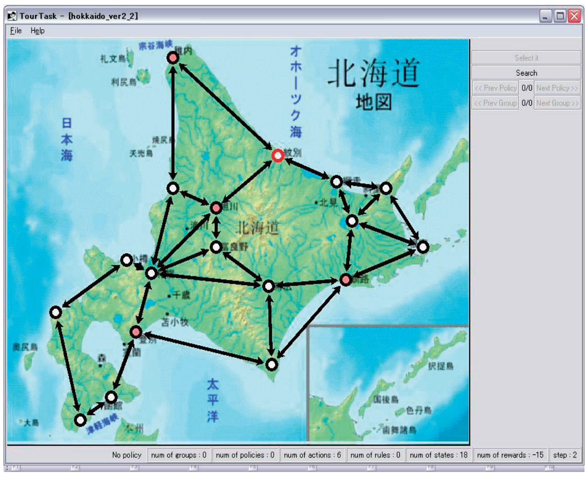
Fig. 9 fine recommendation in the selected group

5. EXPERIMENT

We perform the experiment with twenty subjects to evaluate the effectiveness of our method.

5.1 The round-trip plan task

Fig. 10 shows the round-trip plan task in Hokkaido. The task for a subject is to decide the most preferred round-trip plan after selecting four cities to visit among eighteen cities. The task for the system is to estimate t h e preferable round-trip plans t o each user and to recommend them sequentially.


TourTask - [hokkaido_ver2_2]
File Help
北海道
地図
オホーツク海
日本海
太平洋
Select it
Search
<< Prev Policy 0/0 Next Policy >>
<< Prev Group 0/0 Next Group >>

No policy | num of groups : 0 | num of policies : 0 | num of actions : 6 | num of rules : 0 | num of states : 18 | num of rewards : -15 | step : 2

Fig. 10 The round-trip plan task in Hokkaido

5.2 Experimental results

Table 1 shows the result of the most preferred plans of each twenty subjects. As the experimental results, a large number of subjects prefer each *every-visit-optimal* solution than the optimal solution. On the other hand, a small number of subjects prefer each *every-visit-non-optimal* solution.

Table 1 Most Preferred plans of each twenty subjects

every-visit-solution		*optimal*
every-visit-optimal	*every-visit-non-optimal*	*solution*
10	5	5

5.3 Discussion

(1) Why the end-users' preferences are divided?

We discuss the reason why the end-users' preferences are divided into two groups. According to the results of the questionnaire survey, most of subjects selected *every-visit-optimal* plan have less experience to visit the Hokkaido. In contrast, minor subjects selected *every-visit-non-optimal* plans those have additional cities to visit by the *every-visit-non-optimal* plan recommendation. It suggests that the change of the end-users' preference occurs whether they have the background knowledge of the task or not. Note that in our current plan recommendation system, no background knowledge on the recommended round-trip plan except Fig. 10 is presented to each subject. If any information about recommended plan is provided, we expect that the result on preference change of these two kinds of subjects will differ.

(2) The computing time by interactive LC-learning

The computing time of the round-trip plan task in Fig.10 including graphical output by interactive LC-learning is no more than one second or less per user input, since it is a deterministic MDP model. So we summarize the search cost of LC-Learning in a stochastic case [6].

We compare the search cost of LC-Learning to that of Modified-PIA[5]. Note that Modified-PIA is added preprocess to search *various policies*.

We used MDPs that consist of randomly set state transition probability and reward function for experimental stochastic environment, in which the number of states are 10, the number of actions are 4, and the number of rewards are varied among 1 to 10 to examine the tendency of the search cost when the number of reward is changed.

Fig. 11 shows the comparative search cost when the number of reward is changed. The result indicates that the tendency of search const of Modified-PIA is non-linear and one of LC-Learning is linear when the number of rewards increases. In Modified-PIA, the MDPs those contain the subset of reward set of original MDP are made and the optimal policies for each MDP are searched. So original Modified-PIA is performed 2^r-1 times where r is the number of rewards. After one reward is added, incremental search cost is following.

$$(2^{r+1}-1) - (2^r-1) = 2^r \qquad (4)$$

So it is considered that the search cost of Modified-PIA increases nonlinearly when the number of rewards increases linearly. In contrast, the number of tree structure increase linearly when the number of rewards is increase in LC-Learning. So it is considered that the search cost of LC-Learning increase linearly when the number of rewards increase linearly.

This suggests that our method is better than previous reinforcement learning methods for interactive reinforcement learning in which many rewards are added incrementally.

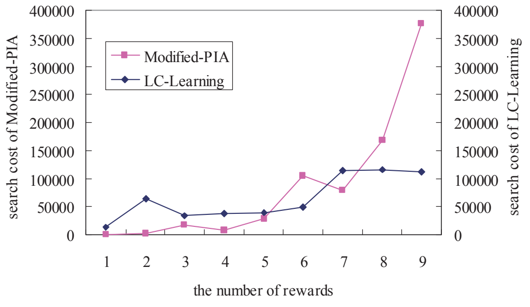


Fig. 11 the search cost when the number of reward is changed [6]

(3) The every-visit optimality in a non-deterministic environment

In a stochastic environment, every-visit optimality is defined as *p-every-visit optimality* where each reward is visited stochastically by not less than probability p (0 < p =< 1). Note that *1-every-visit optimality* is that each reward is visited deterministically even in a stochastic environment.

5. CONCLUSIONS

In this paper, we proposed a new method of recommending preferable solutions of a user in interactive reinforcement learning application task. To cover various preferences of users, we also proposed a concept of *ev-optimality* to define *various policies* and the *coarse to fine recommendation* strategy to guide the most preferable solution of each user.

The future work is to make clear the hypothesis that the change of the end-users' preference occurs whether they have the background knowledge of the task or not. If it true, better recommendation strategies will be selected according to the degree of user's background knowledge of the task.

REFERENCES

[1] Kaplan, F., Oudeyer, P-Y., Kubinyi, E. and Miklosi, A., Robotic clicker training, Robotics and Autonomous Systems, 38-3-4, pp.197-206, 2002

[2] Konidaris, George and Barto, Andrew, Automonous Shaping: Knowledge Transfer in Reinforcement Learning, Proc. of 23rd International Conference on Machine Learning, pp.489-496, 2006

[3] Marthi, Bhaskara, Automatic shaping and decomposition of reward functions, Proc. of 24th international conference on Machine learning, pp.601-608, 2007

[4] Ng, Andrew Y., Harada, Daishi, and Russell, Stuart J., Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping, Proc. of 16th International Conference on Machine Learning, pp.278-287, 1999

[5] Puterman, M.L., Markov Decision Processes: Discrete Stochastic Dynamic Programming, JOHN WILEY & SONS, INC, pp.385-388, 1994

[6] Satoh, Kazuhiro and Yamaguchi, Tomohiro, Preparing various policies for interactive reinforcement learning, SICE-ICASE International Joint Conference 2006, 2006

[7] Konda, Taro, Tensyo, Shinjiro and Yamaguchi, Tomohiro, LC-Learning: Phased Method for Average Reward Reinforcement Learning - Preliminary Results -, PRICAI2002: Trends in Artificial Intelligence, M.Ishizuka and A.Sattar (Eds.), Lecture notes in Artificial Intelligence 2417, Springer, pp.208-217, 2002