

Minimax-Based Reinforcement Learning with State Aggregation

Guofei Jiang and Cang-Pu Wu
Department of Automatic Control, Beijing Institute of Technology
Beijing 100081, P. R. China
email : wucpu@sun.ihep.ac.cn

George Cybenko
Thayer School of Engineering, Dartmouth College
Hanover, NH 03755, USA
email : george.cybenko@dartmouth.edu

Abstract : One of the most important issues in scaling up reinforcement learning for practical problems is how to represent and store cost-to-go functions with more compact representations than lookup tables. In this paper, we address the issue of combining the simple function approximation method—state aggregation with minimax-based reinforcement learning algorithms and present the convergence theory for on-line Q-hat-learning with state aggregation. Some empirical results are also included.

Keywords : reinforcement learning, minimax criterion, dynamic programming, state aggregation

1. Introduction

Associated with dynamic programming (DP) and stochastic approximation theory, a number of DP-based reinforcement learning algorithms[1-4] are developed for solving stochastic optimal control problems recently. One of the most important issues in scaling the reinforcement learning algorithms to these practical problems is how to represent and store cost-to-go functions with more compact representations than lookup tables. Unfortunately a common assumption of the convergence theory for these DP-based learning algorithms is that the cost-to-go functions are represented with lookup tables. It has been shown[1][5-6] that the use of other function approximation methods may damage the convergence properties of these algorithms. In this paper, we address the issue of combining the simple function approximation method—state aggregation with minimax-based reinforcement learning algorithms. Further we present the convergence theory for on-line Q-hat-learning[4] with state aggregation, which is a counterpart to Watkins' Q-learning with regard to minimax criterion. Some empirical results are also included.

2. Minimax-based Markov Decision Tasks and Q-Hat-Learning

Before Q-hat-learning is introduced, a discrete-time Markov decision model is first described. S is the state set of the Markov decision problem(MDP), A is the set of actions available in each state, and $C(C \subset R)$ is the set of immediate costs. Here S, A and C are finite sets. At each time step, the current state i is observed and a decision a is selected from A . After a is performed, the system goes to a next state j with some probability $P_{ij}(a)$. Associated with this state transition, an immediate reward $r \in C$ is gained ($|r| < \infty$ for all i, j and a).

Following the definitions of Heger[7], we define the total discounted reward with respect to a given policy π as $R^\pi = \sum_{k=0}^{\infty} \gamma^k r_k^\pi$, where $\gamma(0 < \gamma < 1)$ is the

discounted factor and r_k^π is the immediate reward with respect to the policy π at time step k . For minimax-based Markov decision tasks, value function V^π is used to measure the performance of a given policy π under the minimax criterion. We define V^π as follows:

$$V^\pi(i) = \sup \left\{ v \in R \mid P(R^\pi > v \mid I_0^\pi = i) > 0 \right\} \quad (1)$$

where I_0^π denotes the starting state. In other words, $V^\pi(i)$ is the worst-case return that can possibly occur if the system starts in the state i and follows the policy π . Further we define the minimax optimal value function $V^*(i) = \inf_{\pi} V^\pi(i)$ for every $i \in S$. Let $c(i, a, j)$ be

the worst immediate cost that can occur in the transition from state i into state j under action a , i.e.

$$c(i, a, j) = \sup\{r \in C | P(i, a, j, r) > 0\} \quad (2)$$

where $P(i, a, j, r)$ denotes the probability that the immediate reward of an episode is r for a given starting state i , action a and successor j . Let $N(i, a)$ be the set of states that immediately can be reached from state i by action a , i.e.

$$N(i, a) = \{j \in S | P(i, a, j) > 0\} \quad (3)$$

where $P(i, a, j)$ denotes the probability that the successor state of an episode is j if action a is executed in starting state i .

With respect to a policy π , the Q value is defined as: $\forall i \in S, a \in A$

$$Q^\pi(i, a) = \max_{j \in N(i, a)} [c(i, a, j) + \gamma V^\pi(j)] \quad (4)$$

where $V^\pi(j) = \min_b Q^\pi(j, b)$. The object in Q-hat-learning is to estimate the minimax optimal Q values that is defined by:

$$Q^*(i, a) = \max_{j \in N(i, a)} [c(i, a, j) + \gamma V^*(j)] \quad (5)$$

where $V^*(j) = \min_b Q^*(j, b)$.

Q-hat-learning is often used on-line, i.e. actual experience interacting with the environment is used to incrementally improve Q values. After each episode k , the Q value associated with the starting state i and executed action a is updated as follows:

$$Q_k(i, a) = \max\{Q_{k-1}(i, a), r_k + \gamma V_{k-1}(j)\} \quad (6)$$

where $V_{k-1}(j) = \min_b Q_{k-1}(j, b)$, the initial Q values $Q_0(i, a)$ for all states and actions are assumed given and satisfy $Q_0(i, a) \leq Q^*(i, a)$.

Heger proved the following convergence theorem for the Q-hat-learning algorithm.

Theorem 1: If every action that is admissible in a state is executed infinitely often in that state, $Q_k(i, a)$ in Equation (6) converges to $Q^*(i, a)$ with probability one for every pair of states and actions as $k \rightarrow \infty$.

3. State Aggregation

Due to the curse of dimensionality, it is often impractical for DP-based learning to compute and store every component of the cost-to-go functions. This limitation can be overcome to some degree by using compact representations to approximate cost-to-go functions, such

as artificial neural networks, polynomials, and decision trees. State aggregation is one of the simplest function approximation methods, which can obviously reduce state space size and accelerate DP-based learning.

The state set of the original Markov decision problem $S = \{s^1, s^2, \dots, s^N\}$ is partitioned into M ($0 < M < N$) clusters S_1, S_2, \dots, S_M with $S = \bigcup_{i=1}^M S_i$ and $S_i \cap S_j = \emptyset$ if $i \neq j$. In the following paragraphs, we use symbol s to represent the individual states of the original MDP and symbol X^i to represent the individual cluster state aggregated from all states $s \in S_i$. Then a cluster state set $\mathcal{X} = \{X^1, X^2, \dots, X^M\}$ is formed. In the original MDP, on-line Q-hat-learning essentially observes from the real system a sequence of quadruples (s_t, a_t, s_{t+1}, r_t) (the time step index $t \in [0, \infty)$) which represent transitions from the current state s_t to the next state s_{t+1} under the current action a_t with an associated immediate reward r_t . Here we assume $s_t \in S_i$ and $s_{t+1} \in S_j$ (here i may be equal to j). Thus according to Equation (6), Q-hat-learning can update Q values as:

$$Q_t(s_t, a_t) = \max\{Q_{t-1}(s_t, a_t), r_t + \gamma V_{t-1}(s_{t+1})\} \quad (7)$$

However, after state aggregation, in the Q-hat-learning based on the cluster state set, only the sequence of quadruples $(X_t^i, a_t, X_{t+1}^j, r_t)$ can be observed, where

X_t^i and X_{t+1}^j are separately mapped from s_t and s_{t+1} . The minimax-based decision process with state aggregation is shown in Figure 1. Just like the learning tasks in partially observable MDPs (POMDPs)[8], the decision problem based on the aggregated states set is essentially non-Markovian because of the hidden states[9]. Then a problem is that if we directly apply the Q-hat-learning algorithm in Equation (8) for this Non-Markovian Decision Problem (NMDP), what does it learn?

$$\bar{Q}_t(X^i, a) = \max\{\bar{Q}_{t-1}(X^i, a), r_t + \gamma \bar{V}_{t-1}(X^j)\} \quad (8)$$

Though state aggregation is mainly used to reduce the computing complexity in DP-based learning, in some real cases it happens naturally, e.g. Due to sensors' limited discrimination, some real states with similar features are naturally clustered into one observed data. Then an agent can only learn and make decisions based on the aggregated data.

4. Convergence Analysis

In the following paragraphs, we prove the convergence

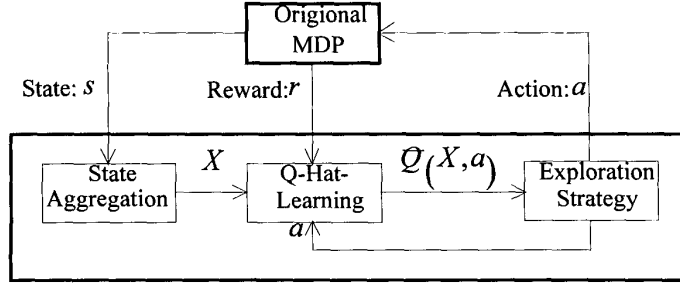


Figure 1: The minimax-based decision process with state aggregation

theory for on-line Q-hat-learning with state aggregation .

Theorem 2: In a minimax-based decision task described above , after state aggregation $\bar{Q}_i(X^i, a)$ in the equation (8) will converge to the solution of the following equation $Q^*(X^i, a)$ with probability one (under the same conditions required for convergence of Q-hat-learning in MDPs). $\forall X^i \in \mathcal{X}, a \in A$,

$$Q^*(X^i, a) = \max_{X^j \in N(X^i, a)} \left\{ c(X^i, a, X^j) + \gamma \min_b Q^*(X^j, b) \right\}, \quad (9)$$

where

$$\begin{aligned} c(X^i, a, X^j) &= \max \left\{ c(s, a, s') \mid s \in S_i, s' \in S_j, P(s, a, s') > 0 \right\}, \\ N(X^i, a) &= \left\{ X^j \in \mathcal{X} \mid s \in S_i, s' \in S_j, P(s, a, s') > 0 \right\}. \end{aligned}$$

Proof : Assume the immediate reward is bounded , i.e. $|r_t| \leq \mathfrak{R}$, \mathfrak{R} is a positive constant . It is easy to show that Q-hat-learning in Equation (8) can converge because for every $X^i \in \mathcal{X}$ and $a \in A$, the Q functions $\bar{Q}_i(X^j, a)$ are monotone increasing with respect to time t and obviously satisfy :

$$\begin{aligned} \bar{Q}_0(X^i, a) &\leq \bar{Q}_{i-1}(X^i, a) \leq \bar{Q}_i(X^i, a) \\ &\leq \sum_{k=0}^{\infty} \gamma^k \mathfrak{R} = \frac{\mathfrak{R}}{1-\gamma} \end{aligned} \quad (10)$$

where $\bar{Q}_0(X^i, a)$ is the initial Q value . For convenience , we denote the Q value which $\bar{Q}_i(X^j, a)$ converges to as $\bar{Q}^*(X^i, a)$.

Now we define a new MDP with the state set \mathcal{X} and

action set A . At each time step , the current state X^i is observed and a decision a is selected from A . After a is performed , the system goes to a next state X^j with a probability $P(X^i, a, X^j)$ ($P(X^i, a, X^j) > 0$) . Associated with this state transition , an immediate reward r ($P(X^i, a, X^j, r) > 0$) is gained . For every $X^i, X^j \in \mathcal{X}$, $a \in A$ and $r \in C$, the probabilities $P(X^i, a, X^j)$ and $P(X^i, a, X^j, r)$ can be defined in any way that satisfy the following constraints :

$$\begin{aligned} P(X^i, a, X^j) &\begin{cases} > 0 & \text{if } \exists s \in S_i, s' \in S_j : P(s, a, s') > 0 \\ = 0 & \text{otherwise} \end{cases} \end{aligned} \quad (11)$$

$$\begin{aligned} P(X^i, a, X^j, r) &\begin{cases} > 0 & \text{if } \exists s \in S_i, s' \in S_j : P(s, a, s', r) > 0 \\ = 0 & \text{otherwise} \end{cases} \end{aligned} \quad (12)$$

For convenience , in the following we will denote the new defined MDP as AMDP . Obviously we can establish many possible AMDPs that satisfy the above constraints , e.g. an AMDP can be defined with the following determined state transition probabilities

$$\begin{aligned} P(X^i, a, X^j) &= \begin{cases} 1/N(X^i, a) & \text{if } \exists s \in S_i, s' \in S_j : P(s, a, s') > 0 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (13)$$

Now if we apply Q-hat-learning for anyone of these AMDPs , according to theorem 1 , the Q values $Q_i(X^i, a)$ of this AMDP will converge to a unique optimal Q value $Q^*(X^i, a)$, which is the solution of Equation (9) .

By induction on time step t , we will show in the following that the optimal Q values of the NMDP is

equal to the optimal Q values of the AMDPs , i.e. $\bar{Q}^*(X,a)=Q^*(X,a)$. Now we apply Q-hat-learning for the NMDP as well as an AMDP and observe two infinite sequences of quadruples (X_t, a_t, X_{t+1}, r_t) . In the following we show two important facts about these two sequences :

Fact 1 : If every state-action pair (X,a) is updated infinitely often in learning (the conditions required for convergence of Q-hat-learning in MDPs) , then every quadruple (X,a,X',r) with $P(X,a,X') > 0$ and $P(X,a,X',r) > 0$ will also appear infinitely often in the sequences .

Fact 2 : Because the AMDPs satisfy the constraints (11) and (12) , it is straightforward to show that every kind of quadruples in the sequence of an AMDP will also appear in the sequence of the NMDP and vice versa . However the same quadruples may appear in two sequences at different time step and with different frequency because of the different state transition probabilities and reward probabilities .

Assume the initial Q values of the NMDP and AMDP are same , i.e. $\bar{Q}_0(X,a)=Q_0(X,a)$ for every $X \in \mathcal{X}$ and $a \in \mathcal{A}$. Here we separately use t and k to denote the time step indexes of the sequences of the NMDP and AMDP and assume $t = 0 \Leftrightarrow k = 0$. Now we select a sequence of the NMDP as a reference sequence and by induction on NMDP's time step t , we will show $\bar{Q}^*(X,a) \leq Q^*(X,a)$ for every $X \in \mathcal{X}$ and $a \in \mathcal{A}$. Assume at time step $t = 0$ the first quadruple in the NMDP's sequence is (X'_0, a'_0, X'_1, r'_0) . Because of the Fact 2 , it is certain that among $k \in [0, \infty)$ we can find a time step k_0 in the AMDP's sequence where the quadruple $(X_{k_0}, a_{k_0}, X_{k_0+1}, r_{k_0})$ is same with the quadruple (X'_0, a'_0, X'_1, r'_0) . Since the Q functions $Q_t(X,a)$ in Q-hat-learning are monotone increasing with respect to time t , it is obvious to show that for every $X \in \mathcal{X}$ and $a \in \mathcal{A}$,

$$Q_{k_0}(X,a) \geq Q_0(X,a) = \bar{Q}_0(X,a) \quad (14)$$

At time step $t = N$ the observed quadruple in the NMDP's sequence is $(X'_N, a'_N, X'_{N+1}, r'_N)$. Because of Fact 1 and Fact 2 , similarly among $k \in (k_{N-1}, \infty)$ we can find a time step k_N in the AMDP's sequence where the quadruple is same with the quadruple $(X'_N, a'_N, X'_{N+1}, r'_N)$. Now for time step $t = N$, we

assume for every $X \in \mathcal{X}$ and $a \in \mathcal{A}$ the following inequality holds

$$Q_{k_N}(X,a) \geq \bar{Q}_N(X,a) \quad (15)$$

At time step $t = N + 1$, the observed quadruple in the NMDP's sequence is $(X'_{N+1}, a'_{N+1}, X'_{N+2}, r'_{N+1})$. Similarly among $k \in (k_N, \infty)$ we can find a time step k_{N+1} in the AMDP's sequence where the quadruple $(X_{k_{N+1}}, a_{k_{N+1}}, X_{k_{N+1}+1}, r_{k_{N+1}})$ is same with the quadruple $(X'_{N+1}, a'_{N+1}, X'_{N+2}, r'_{N+1})$, i.e. $X_{k_{N+1}} = X'_{N+1}$, $a_{k_{N+1}} = a'_{N+1}$, $X_{k_{N+1}+1} = X'_{N+2}$ and $r_{k_{N+1}} = r'_{N+1}$.

The NMDP's Q values $\bar{Q}_{N+1}(X,a)$ are updated in learning according to the following equation (16) :

$$\begin{aligned} & \text{If } (X,a) = (X'_{N+1}, a'_{N+1}), \\ & \bar{Q}_{N+1}(X,a) \\ & = \max\left(\bar{Q}_N(X,a), r'_{N+1} + \gamma \min_b \bar{Q}_N(X'_{N+2}, b)\right); \\ & \text{If } (X,a) \neq (X'_{N+1}, a'_{N+1}), \\ & \bar{Q}_{N+1}(X,a) = \bar{Q}_N(X,a) . \end{aligned} \quad (16)$$

On the other hand , the AMDP's Q values $Q_{k_{N+1}}(X,a)$ are updated in learning according to the following equation (17) :

$$\begin{aligned} & \text{If } (X,a) = (X_{k_{N+1}}, a_{k_{N+1}}), \\ & Q_{k_{N+1}}(X,a) = \\ & \max\left(Q_{k_{N+1}-1}(X,a), r_{k_{N+1}} + \gamma \min_b Q_{k_{N+1}-1}(X_{k_{N+1}+1}, b)\right) \\ & \text{If } (X,a) \neq (X_{k_{N+1}}, a_{k_{N+1}}), \\ & Q_{k_{N+1}}(X,a) = Q_{k_{N+1}-1}(X,a) . \end{aligned} \quad (17)$$

According to the definition of k_N and k_{N+1} , obviously we have

$$k_{N+1} - 1 \geq k_N \quad (18)$$

By Inequality (15) , (18) and the Q values' monotone increasing property , we have

$$Q_{k_{N+1}-1}(X,a) \geq Q_{k_N}(X,a) \geq \bar{Q}_N(X,a) \quad (19)$$

for every $X \in \mathcal{X}$ and $a \in \mathcal{A}$. Further we can obtain

$$\min_b Q_{k_{N+1}-1}(X,b) \geq \min_b \bar{Q}_N(X,b) \quad (20)$$

for every $X \in \mathcal{X}$. For state-action pairs $(X,a) = (X'_{N+1}, a'_{N+1}) = (X_{k_{N+1}}, a_{k_{N+1}})$, by Inequality (19) , (20) and $r_{k_{N+1}} = r'_{N+1}$, $X_{k_{N+1}+1} = X'_{N+2}$, we have

$$\begin{aligned}
Q_{k_{N+1}}(X, a) &= \max \left\{ Q_{k_{N+1}-1}(X, a), \right. \\
&\quad \left. r_{k_{N+1}} + \gamma \min_b Q_{k_{N+1}-1}(X_{k_{N+1}+1}, b) \right\} \\
&\geq \max \left(\bar{Q}_N(X, a), r'_{N+1} + \gamma \min_b \bar{Q}_N(X'_{N+2}, b) \right) \\
&= \bar{Q}_{N+1}(X, a) \tag{21}
\end{aligned}$$

For state-action pairs $(X, a) \neq (X'_{N+1}, a'_{N+1}) = (X_{k_{N+1}}, a_{k_{N+1}})$, by Equation (16), (17) and Inequality (15) we have

$$\begin{aligned}
Q_{k_{N+1}}(X, a) &= Q_{k_{N+1}-1}(X, a) \geq Q_{k_N}(X, a) \\
&\geq \bar{Q}_N(X, a) = \bar{Q}_{N+1}(X, a) \tag{22}
\end{aligned}$$

Further by Inequality (21) and (22), we can conclude that for the time step $t = N + 1$ the following inequality holds for every $X \in \mathcal{X}$ and $a \in A$

$$Q_{k_{N+1}}(X, a) \geq \bar{Q}_{N+1}(X, a) \tag{23}$$

Thus for every NMDP's time step t , in the same way we can find a time step k_t in the AMDP's sequence where the Q values satisfy the following inequality for every $X \in \mathcal{X}$ and $a \in A$

$$Q_{k_t}(X, a) \geq \bar{Q}_t(X, a) \tag{24}$$

As shown above, $\bar{Q}_t(X, a)$ and $Q_{k_t}(X, a)$ separately converge to $\bar{Q}^*(X, a)$ and $Q^*(X, a)$ as $t \rightarrow \infty$ (obviously $k_t \rightarrow \infty$ as $t \rightarrow \infty$). Then by Inequality (24), we have

$$Q^*(X, a) \geq \bar{Q}^*(X, a) \tag{25}$$

for every $X \in \mathcal{X}$ and $a \in A$.

On the other hand, if we select a sequence of the AMDP as a reference sequence and by induction on AMDP's time step k , in the same way we obtain the following inequality

$$Q^*(X, a) \leq \bar{Q}^*(X, a) \tag{26}$$

for every $X \in \mathcal{X}$ and $a \in A$.

Thus by Inequality (25) and (26), we have

$$Q^*(X, a) = \bar{Q}^*(X, a) \tag{27}$$

for every $X \in \mathcal{X}$ and $a \in A$ which verifies the theorem 2 and the convergence of Q-hat-learning with state aggregation is proved. **Q.E.D.**

Though $\bar{Q}_t(X^i, a)$ converges to $Q^*(X^i, a)$, it is easy to give examples to illustrate that $Q^*(X^i, a)$ may differ greatly from the correct values $Q^*(s, a)$ for some $s \in S_i$. Define $\bar{V}^*(X^i) = \min_{a \in A} \bar{Q}^*(X^i, a)$ and

$V^*(s) = \min_{a \in A} Q^*(s, a)$. Now if we aggregate states based on their similar features and assume after state aggregation we have $\max_{S_i \subset S} \max_{s, s' \in S_i} |V^*(s) - V^*(s')| \leq \varepsilon$,

then what will be the error bound

$\max_{X^i \in \mathcal{X}} \max_{s \in S_i} |V^*(X^i) - V^*(s)|$? Unfortunately, till now

we have no conclusion about this error bound. However, the experiment in the next section shows that if the states of some problems are aggregated appropriately, $V(X^i)$ can approximate $V(s)$ well for every $s \in S_i$.

5. Examples

A machine replacement problem is a typical example of a wide range of replacement problems in industry. In this paper, we choose the automobile replacement problem in [10] as our example. However, here the replacement problem is based on the minimax decision criterion. For convenience, we consider the automobile replacement problem within ten years. Every three months we observe the states once and make decisions whether keep the current automobiles or replace them. Here the system state is represented by the automobiles age which increases one after a period of three months, then the states range from 1 to 40. Moreover let the age of the automobile aged 40 still be 40 in the future so as to keep the number of states finite.

There are total 41 feasible decisions available for every state. Under the first decision ($k=1$), the current automobile is kept during the next quarter; Under other decisions ($k>1$), the current automobile is sold and another automobile aged $k-2$ is bought. Then we have total 40 states and 41 decisions available for every state so that there are total 41^{40} admissible policies for this problem. See more details in [10].

Now we separately aggregate the 40 states into 20 and 10 clusters. According to the structure of the problem, the consecutive two (or four) states are mapped into one cluster state because in this problem the consecutive states obviously have more similar features. Here we define

$$Error = \max_{X^i \in \mathcal{X}} \max_{s \in S_i} \left| \frac{V(X^i) - V(s)}{V(s)} \right| \times 100\% \quad , \quad \text{where}$$

$V(s) = \min_{b \in A} Q(s, b)$ is the optimal value of state s and

$V(X^i) = \min_{b \in A} Q(X^i, b)$ is the optimal value of the cluster state.

Error is used to show how closely $V(X^i)$ can approximate the correct value $V(s)$ for all

$s \in S_i$. *Iteration steps* in the following table refers to the number of Q value updates before convergence. Now

we apply Q-hat-learning for the decision problem with different state aggregation and get the results of experiment shown in Table 1 .

Table 1: The results comparison for different states aggregation

Number of States	40 states	20 clusters	10 clusters
Error (%)	0%	2.50%	5.87%
Iteration Steps	932,125	345,321	149,532

In our experiment , the semi-uniform distributed exploration strategy is used for Q-hat-learning , i.e. with probability p the action that maximizes the Q-value is selected and with probability $1 - p$ all feasible actions are randomly selected . Let all initial Q values equal to a same value .

The results of our experiment show that while the states space is aggregated into more small cluster space , the approximation error will increase and the learning based on the mapped cluster space will become obviously faster .

6. Conclusions

This paper is motivated by the desire to use state aggregation method to reduce states space size and accelerate DP-based learning . We discussed the state aggregation issues in the on-line minimax-based reinforcement learning and proved the convergence of Q-hat-learning with state aggregation . An example is included to show that for some problems , if we aggregate states appropriately, the cost-to-go functions can be approximated well by state aggregation method .

Acknowledgments

The work of our paper is supported by the National Natural Science Foundation of China .

References

- [1] R.S.Sutton, "Learning to Predict by the Methods of Temporal Differences" , *Machine Learning* , vol. 3 , 1988 , pp. 9-44 .
- [2] C.J.C.H.Watkins, "Learning from Delayed Rewards," *Ph.D.Dissertation* , Cambridge University , UK , 1989.
- [3] A.G.Barto, S.J.Bradtko, and S.P.Singh, "Learning to Act Using Real-Time Dynamic Programming ," *Artificial Intelligence* , vol. 72 , 1995 , pp. 81-138 .
- [4] M.Heger, "Consideration of Risk in Reinforcement Learning," *Proceedings of the Eleventh International Machine Learning Conference* , Morgan Kaufmann, 1994 .
- [5] G.J.Gordon, " Stable Function Approximation in

Dynamic Programming," *Proceedings of the Twelfth International Machine Learning Conference* , Morgan Kaufmann, 1995 .

[6] J.A.Boyan, A.W.Moore, "Generation in Reinforcement Learning: Safely Approximation the Value Function," *Advances in Neural Information Processing Systems 7* , Morgan Kaufmann ,1995 .

[7] M.Heger, "The Loss from Imperfect Value Function in Expectation-Based and Minimax-Based Tasks," *Machine Learning* , vol. 22 , 1996 , pp. 197-225 .

[8] S.P. Singh, T.Jaakkola, and M.I.Jordan, "Learning without State Estimation in Partially Observable Markovian Decision Process, " *Proceedings of the Eleventh International Machine Learning Conference* , Morgan Kaufmann ,1994 .

[9] S.D.Whitehead, L.J.Lin, "Reinforcement Learning of non-Markov Decision Processes ," *Artificial Intelligence* , vol. 73 , 1995 , pp. 271-306 .

[10] R.A.Howard, *Dynamic programming and Markov processes* , M.I.T. Press,1960.