Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 120760, 6 pages
http://dx.doi.org/10.1155/2014/120760

Research Article

A Reward Optimization Method Based on Action Subrewards in Hierarchical Reinforcement Learning

Yuchen Fu,[1,2] Quan Liu,[2] Xionghong Ling,[2] and Zhiming Cui[2]

[1] *Suzhou Industrial Park Institute of Services Outsourcing, Suzhou, Jiangsu 215123, China*
[2] *School of Computer Science and Technology, Soochow University, Suzhou, Jiangsu 215006, China*

Correspondence should be addressed to Yuchen Fu; yuchenfu68@gmail.com

Received 15 August 2013; Accepted 14 November 2013; Published 28 January 2014

Academic Editors: J. Shu and F. Yu

Reinforcement learning (RL) is one kind of interactive learning methods. Its main characteristics are "trial and error" and "related reward." A hierarchical reinforcement learning method based on action subrewards is proposed to solve the problem of "curse of dimensionality," which means that the states space will grow exponentially in the number of features and low convergence speed. The method can reduce state spaces greatly and choose actions with favorable purpose and efficiency so as to optimize reward function and enhance convergence speed. Apply it to the online learning in Tetris game, and the experiment result shows that the convergence speed of this algorithm can be enhanced evidently based on the new method which combines hierarchical reinforcement learning algorithm and action subrewards. The "curse of dimensionality" problem is also solved to a certain extent with hierarchical method. All the performance with different parameters is compared and analyzed as well.

1. Introduction

Reinforcement learning (RL) is one kind of interactive learning methods. Its main characteristics are "trial and error" and "related reward." In the whole learning process, the agent can communicate with environment to get rewards and improves actions according to these rewards. Repeat the process over enough times, an optimal action can be gained, and all the optimal actions compose optimal policies.

Q-learning, which tries to map every state-action pair into a real number (Q-value), is one of the basic and classical methods in reinforcement learning field. But it is only suitable for small state spaces. When state spaces become larger or even infinite and continuous, it is not effective to visit and store the same states time after time. Then there arose the "curse of dimensionality" problem. Recently, the main ideas used to solve this problem are abstraction and approximation. All these ideas can be divided into four kinds of methods: spaces clustering algorithms, finite spaces search methods, value functions approximation methods, and hierarchical reinforcement learning (HRL) [1]. A reinforcement learning method based on node-growing k-means clustering algorithm is described in the literature [2], which is used to solve the adaptive partition problem in continuous state spaces. There are also many function estimate methods in value function approximation, such as neural network and kernel methods [3, 4]. The relational reinforcement learning (RRL) method, which had been researched and expatiated in the literature [5–7], has been concerned most in recent years. In order to get a further purpose, other methods such as Gaussian processes, graphical kernels, and logical algorithm had been combined with RRL as well [8–10]. At present, the achievements of HRL method include Option [11], MAXQ [12], and HAM [13]. These methods decompose the learning task into a hierarchy of smaller subtask. Thus, in HRL method, if the agent is only concerned about the current local space change or the subtarget state change, the update process of the policy will be restricted in the local space or high-level space, so that you can speed up the learning process and reduce the dependence of the algorithm's convergence speed with the environmental changes.

At present, the Option and MAXQ of the HRL algorithms have been widely used. They have proved that they not only can be applied to the task which has huge state space, but also can apply the experience to other similar learning tasks, greatly accelerating the learning speed. Schultink

et al. proposed an EHQ algorithm based with MAXQ. This algorithm is, in theory, able to converge to the hierarchical optimal, compared with MAXQ which can only converge to the recursively optimal policy [14]. To use the HRL method, we must first solve the two problems: (1) search the subgoal of the subtask and (2) decompose the task automatically. Many ways have been offered to the search for the subgoal. All these methods are based on the experience gained in the past to get the subgoals, which can be divided into two types; one is based on the statistical characteristics of the state access frequency to find subgoals.

Representative achievements are as follows: Mannor et al. think that states serve as potential subgoals if they are frequently visited on successful paths but are not on unsuccessful visited ones [15]; Stolle and Precup use the nodes visited most frequently as a subgoal [16]. In these two approaches, sometimes the results obtained are not very satisfactory, because the state which is around the real subgoals state may also be visited many times; Şimşek et al. find the subgoals by dividing the recent experience of local state transition diagram [17]; Subgoals' finding is ultimately server for decomposing the task automatically.

Because of the "curse of dimensionality" problem in most reinforcement learning tasks and the low convergence speed of traditional algorithms, it is important to improve both of them in RL methods. The key problems solved in this paper are "curse of dimensionality" problems in large discrete state spaces of infinite repetition tasks and the reward functions optimization in convergence speed enhancement of algorithms such as Q-learning. At the same time, action subrewards are proposed to evaluate the efficiency of actions for achieving the goal. Combined with a hierarchical reinforcement learning method in large discrete state spaces of infinite repetition tasks, it is applied to the Tetris game as an experiment. The experiment result shows that both of the problems mentioned above can be solved to a certain extent.

2. Reinforcement Learning

Reinforcement learning is a concept that parallels with supervised learning and unsupervised learning. The main difference between them is that, in RL, the agent is not told by positive and negative examples, but it uses trial-and-error method to find optimal policies instead. The whole learning process is evolved from ideas in psychology. The agent gets rewards form the interaction with its environment so as to enhance or weaken execution of actions for getting the optimal policies. So the final purpose of reinforcement learning is to maximize the accumulative reward, and the learning framework is showed in Figure 1.

2.1. Markov Decision Processes (MDP). There exist many kinds of modeling methods for environment in reinforcement learning. For stochastic, discrete states and discrete time tasks, it used to take the Markov model. The task in this paper is with discrete states and time, and the states space is huge and stochastic, so the learning process can be modeled as Markov decision processes (MDP). There gives the formalized definition [18, 19].

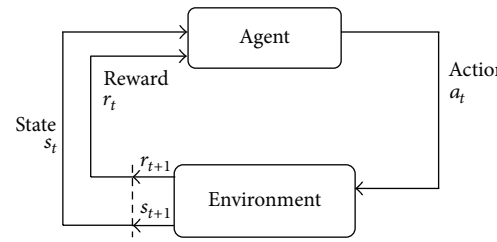

Figure 1: The framework of reinforcement learning.

Markov Decision Processes. It is defined by a tuple $\langle S, A, R, P\rangle$, which contains an environment states set S, an actions set A, a reward function: $R : S\times A \rightarrow R$, and a state transition function: $P : S \times A \rightarrow PD(S')$. $R(s, a, s')$ is the immediate reward in state s under action a transited to state s'. $P(s, a, s')$ is the transition probability in state s under action a to state s'. Laconically, both of the variables can be written as $R^a_{ss'}$ and $P^a_{ss'}$.

Based on the above definition, the key work of reinforcement learning is to solve learning problem with unknown $R^a_{ss'}$ and $P^a_{ss'}$. So the only experience we can use is immediate rewards, which are contacted with state value functions and policies to choose the right policy. And it can be gained that

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = r_{t+1} + R_{t+1}. \tag{1}$$

So, for the special policy, the Bellman equation is as follows:

$$\begin{aligned} V^\pi(s) &= E_\pi\{R_t \mid s_t = s\} \\ &= E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1} \mid s_t = s\right\} \\ &= E_\pi\left\{r_{t+1} + \gamma\sum_{k=0}^{\infty}\gamma^k r_{t+k+2} \mid s_t = s\right\} \\ &= \sum_a \pi(s,a)\sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma E_\pi \right. \\ &\qquad \left. \times\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+2} \mid s_{t+1} = s'\right\}\right] \\ &= \sum_a \pi(s,a)\sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma V^\pi\left(s'\right)\right]. \end{aligned} \tag{2}$$

According to the optimal Bellman equation, the value function of optimal policy is defined as

$$\begin{aligned} V^*(s) &= \max_{a\in A(s)} E\{r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s, a_t = a\} \\ &= \max_{a\in A(s)} \sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma V^*\left(s'\right)\right]. \end{aligned} \tag{3}$$

Solve the above Bellman equation, and we can get the solution for MDP.

2.2. Hierarchical Reinforcement Learning. The hierarchical reinforcement learning (HRL) is proposed by many researchers to solve the "curse of dimensionality" problem. Its essential is adding abstraction to the reinforcement learning framework and dividing a whole task into many different subtasks on different hierarchy, which makes it possible to solve the task in smaller substate spaces, and, as a result, the convergence speed can be enhanced as well. From another point of view, this can be seen as a method for dimensionality reducing, which is good for the solution of problems. Main techniques for HRL include state space decomposition, temporal abstraction, and state abstraction. Classical algorithms are Option, HAM and MAXQ, which considered the hierarchy from angles of action, policy, and task [1]. These are methods researched and applied widely in recent years.

State space decomposition is a method which decomposes the state space into different subsets and then gets solutions with the divide and rule policy. As a result, all the problems can be resolved in small subspaces. Temporal abstraction is a method that groups action sequences and action sets, which divides a single step into many steps so as to decrease the decision-making number and reduce the learning press. State abstraction method, which ignores variables that are irrelative to subtasks, achieves the effect of dimension reduction.

3. Function Optimization Based on Subrewards in Hierarchical RL

3.1. A Hierarchical Method in Infinite Repetition Tasks. On the basis of the theory about reinforcement learning and HRL mentioned in Section 2, a hierarchical method in infinite repetition tasks is proposed. According to the characteristic of tasks in this paper, the state space is decomposed to reduce dimensionality and speed up convergence.

The infinite repetition task M can be decomposed into a task set $\{M_0, M_1, \ldots, M_n\}$, and because of the repetition of the task, the policy π can be decomposed into a policy set $\{\pi_0, \pi_1, \ldots, \pi_n\}$, where π_i is the corresponding policy to M_i and each π_i is parallel. Then the set of each optimal policy π_i^* in subtask M_i is $\pi^* = \{\pi_0^*, \pi_1^*, \ldots, \pi_n^*\}$ which is an optimal policy of the whole task. To some extent, all the subtasks are independent of each other.

Let a subtask M_i be defined by a couple $\langle \pi_i, R_i \rangle$, where π_i is a policy for a subtask and R_i is a reward for a subtask. Let $V^\pi(i, s)$ be the expect reward in subtask M_i with policy π_i, and the corresponding Bellman equation is as follows:

$$V^\pi(i,s) = V^\pi\left(\pi_i(s), s\right) + \sum_{s'} P_i^\pi\left(s \longrightarrow s' \mid s, \pi_i(s)\right) \gamma V^\pi\left(i, s'\right). \tag{4}$$

The hierarchical idea in the infinite repetition task is much more compact than that in the MAXQ method. Because of the repetition, it is easy for us to handle states and actions, and policies can be utilized time after time as well.

3.2. Action Subreward. The exploration and exploitation of actions are always key problems in reinforcement learning, and they are also important points in the solution of convergence speed. Traditional RL methods such as Q-learning could not find a balance between them, and there also exists the low convergence speed problem; that is, the agent has to visit the same states over and over again to achieve the goal for rewards. The larger the state spaces become, the lower the convergence speed would be. There needs to be some efficient way to optimize the reward functions to handle these problems and enhance the convergence speed.

A method based on subrewards is proposed in this paper, which considers the efficiency of actions along with their rewards. The estimate standard is whether actions are good for goal achieving or not. It makes for action choice and resolves the balance between action exploration and exploitation as well. If state values weigh the action quality in current state, then the action subrewards stand for the detailed effect of the action in current state. All these make the learning algorithm more flexible, optimize reward functions, and enhance convergence speed.

Definition 1. The action subreward $A_a(s)$ denotes the efficiency value for agent implements action a to achieve goal in state s, instead of the direct reward it gets.

For a special task, let characteristic set T that affects goal achieving be $\{t_0, t_1, \ldots, t_n\}$, and then the action subreward in state s with action a in that task can be represented as follows:

$$A_a(s) = \alpha_0 t_0 + \alpha_1 t_1 + \cdots + \alpha_n t_n = \sum_{t_i \in T} \alpha_i t_i, \tag{5}$$

where α_i $(-1 \le \alpha_i \le 1)$ is the proportion parameter of each characteristic, and it can be confirmed by its effect in goal achieving. It can be a prize which is good for goal achieving or a punishment that gets the opposite effect.

3.3. Reward Function Optimization Based on Action Subrewards in Infinite Repetition Tasks. According to the theory mentioned in Sections 3.1 and 3.2, the reward function optimization based on action subrewards in infinite repetition tasks can be confirmed now, which means that some modification can be done to (4). It should be divided into two aspects:

(1) there are still many states in subtasks, in which discount reward model is adopted, and its equation can be modified as

$$\begin{aligned} V^\pi(i,s) &= V^\pi\left(\pi_i(s), s\right) + \sum_{s'} P_i^\pi\left(s \longrightarrow s' \mid s, \pi_i(s)\right) \gamma V^\pi\left(i, s'\right) + A_a(s) \\ &= V^\pi\left(\pi_i(s), s\right) + \sum_{s'} P_i^\pi\left(s \longrightarrow s' \mid s, \pi_i(s)\right) \gamma V^\pi\left(i, s'\right) + \sum_{t_i \in T} \alpha_i t_i; \end{aligned} \tag{6}$$

Repeat {every sub-task}
  Initialize all the Q-value
    Repeat {every episode}
      Initialize state $s0$
        Repeat {every step of an episode}
          If the goal can be achieved
            Then select current action
          Else
            Select actions according (5)
            Observe immediate reward r and the next state s'
          Calculate
            $V^{\pi}(i,s) = V^{\pi}(\pi_i(s), s) + \sum_{s'} P_i^{\pi}(s \rightarrow s' \mid s, \pi_i(s))\gamma V^{\pi}(i,s') + A_a(s)$
          Or
            $V^{\pi}(i,s) = V^{\pi}(\pi_i(s), s) + \sum_{s'} P_i^{\pi}(s \longrightarrow s' \mid s, \pi_i(s)) V^{\pi}(i,s') + A_a(s)$
          as the basis of the next action choice
        until s_i is the terminal state
    until the episode is end
until one sub-task is end and then turn to the next

Algorithm 1

(2) there are a few states in subtasks, in which finite-horizon model is taken, and the equation can be modified as

$$\begin{aligned} V^{\pi}(i,s) &= V^{\pi}\left(\pi_i(s), s\right) \\ &\quad + \sum_{s'} P_i^{\pi}\left(s \longrightarrow s' \mid s, \pi_i(s)\right) V^{\pi}\left(i, s'\right) + A_a(s) \\ &= V^{\pi}\left(\pi_i(s), s\right) \\ &\quad + \sum_{s'} P_i^{\pi}\left(s \longrightarrow s' \mid s, \pi_i(s)\right) V^{\pi}\left(i, s'\right) + \sum_{t_i \in T} \alpha_i t_i. \end{aligned} \tag{7}$$

From (6) and (7), it can be concluded that if actions a_i and a_j are taken in a special state, the rewards got from them will be indistinctive in nonterminal states or similar states. This needs the agent to balance the exploitation and exploration of actions, which will spend too much time and calculation to finish this work. But with action subrewards, a_i and a_j can be evaluated in detail to get further rewards and make reasonable decisions. At the same time, all the proportion parameters can be adjusted from a microcosmic angle, and it is more flexible for action choice control.

3.4. Reward Function Optimization Algorithm. On the basis of the theory discussed above, the framework of reward function optimization algorithm is as in Algorithm 1.

For the convergence of this algorithm, what we need to discuss is only the posterior part of (6) and (7), because the $V^{\pi}(\pi_i(s), s) + \sum_{s'} P_i^{\pi}(s \rightarrow s' \mid s, \pi_i(s)) V^{\pi}(i, s')$ part has been proved to be convergent in other literature (please turn to interrelated papers).

Proof. Let Δ be the most error between $V^{\pi}(i,s)$ and $V^{\pi}(i,s')$; that is,

$$\Delta = \max \left| V^{\pi}\left(i, s'\right) - V^{\pi}(i,s) \right|. \tag{8}$$

Because $-1 \leq \alpha_i \leq 1$ in $\sum_{t_i \in T} \alpha_i t_i$, whether it is $\alpha_i \geq 0$ or $\alpha_i \leq 0$, that will be good for choosing the optimal actions, and the speed of $\Delta \rightarrow 0$ is enhanced; that is, the $\sum_{t_i \in T} \alpha_i t_i$ part quickens the convergence speed.

The proof of (7) is the same as (6), and it will not give unnecessary details here. □

3.5. Action Subrewards with Divide and Rule. In order to increase flexibility of the algorithm, considering the changes in different periods, the idea of divide and rule is introduced in this paper. It is rooted in hierarchical method. Because of the diversification in tasks, although there is only one goal in a task, factors that impact goal achieving will change a lot in different periods, which changes the efficiency of action subrewards as well. And then the idea of divide and rule is introduced. It is reasonable and flexible to adjust proportion parameters in time for action selecting when the environment changes. This kind of idea is similar to the hierarchical concept, and it can be understood as a hierarchical method on the action subrewards.

This theory is validated by the experiment in this paper. The result shows that the performance can be improved with such a method.

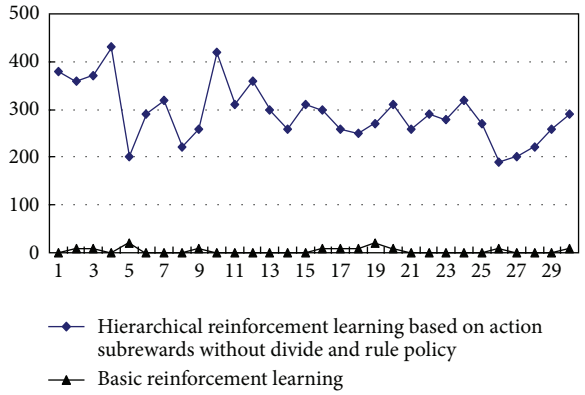

Figure 2: The performance comparison between hierarchical reinforcement learning based on action subrewards without divide and rule policy and basic reinforcement learning.

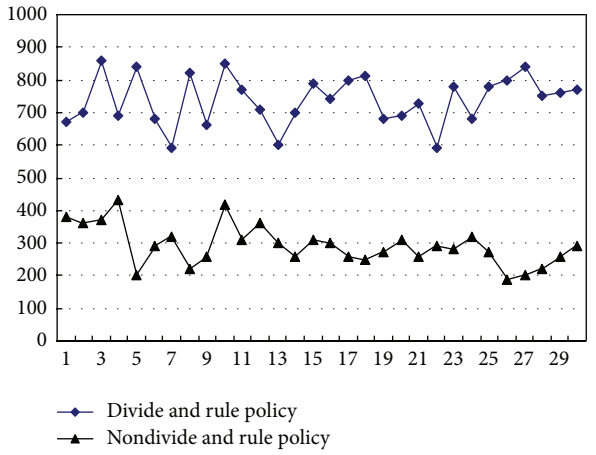

Figure 3: The performance comparison between divide and rule policy and nondivide and rule policy.

4. Experiment and Analyses

The Tetris game was developed by Alexey Pathitov in 1985. Because of its characteristic, it has been a classical problem for large discrete state spaces in reinforcement learning. At the same time, the Tetris game is an infinite repetition task, which needs the player to keep the game continuing and get as many scores as possible. All these factors are the reason why we took the Tetris game as our experiment. In a standard game platform, the game ground is a matrix with 10 columns and 20 rows, there are seven kinds of blocks, most of which have four directions, and the number of states will be 10^{60}, which is a large discrete state space problem for both store and calculation to computers. The Tetris game is exactly a typical representation of the "curse of dimensionality" problem.

The Tetris game is an online game that has high demands upon calculating speed and convergence speed. All of that can be handled with methods proposed in this paper. In the experiment, a standard Tetris game platform is selected (i.e., a 10×20 matrix and seven kinds of blocks), and we can gain 10 points after one line is filled up. All the points we get are nonspecial; that is, the points got from lines filled up one by one are equal to as many lines filled up once. The action set contains turn left, turn right, circumrotate, and go down. It should be noted that because of the small number of states in subtasks after hierarchical handling, the finite-horizon model is employed here. The set of characteristics that affect goal achieving (it is get scores in this experiment) is $T = \{\text{height}, \text{hole}, \text{channel}\}$, and the corresponding proportion parameters are α, β, and γ, which are counterparts of α_1, α_2, and α_3 in (5).

Figure 2 shows the performance compared, between hierarchical reinforcement learning based on action subrewards without divide and rule policy and basic reinforcement learning, where proportion parameters are set as $\alpha = 0.1$, $\beta = 0.2$, $\gamma = 0.1$. The y-axis denotes the number of lines filled up, and the abscissa denotes the number of training steps.

It is easy to conclude that the learning efficiency and outcome of hierarchical RL based on action subrewards are much better than that of the basic RL method. It is obvious that the basic RL method is hard to converge because of the large state spaces in most instances. But after combined action subrewards and hierarchical method, the scores increase obviously. The real-time require is satisfied as well. Even if there are no prior knowledge and training, it works well.

The divide and rule policy that adjusts proportion parameters of α, β, and γ is introduced in this experiment to make the learning more flexible, which makes action subrewards more reasonable. And then action subrewards can reflect action quality in time. Figure 3 shows the performance comparison between divide and rule policy and nondivide and rule policy, where exist two situations according to the height of blocks: $\alpha = 0.1$, $\beta = 0.2$, and $\gamma = 0.1$ as well as $\alpha = 0.2$, $\beta = 0.1$, and $\gamma = 1$; that is, when the height is low, holes could be considered more than height, and height should be considered more with it growing. So policy changes with the environment, which is according to the reality. The result shows that the performance of algorithm improves a lot and scores are much higher.

5. Conclusions

Reinforcement learning which is a method that is independent of supervised learning or unsupervised learning has been studied and applied widely. But there is "curse of dimensionality" problem in basic RL methods, and it is so hard to avoid and conquer, so many solutions had been proposed to resolve this kind of problem.

Aiming at solving "curse of dimensionality" problem and the low convergence speed in RL, a hierarchical RL method based on action subrewards is proposed. Apply it to the Tetris game, and the outcome shows that it is efficient in getting over not only the "curse of dimensionality" problem but also the difficulty of low convergence speed. The whole effect of the experiment has been improved much as well.

But there would be some local optimization problem in this experiment because of the hierarchical method. And we need to do further research on the choice of proportion parameters to make sure the algorithm is more efficient. Both

of the problems mentioned above will be our key work in the future, and there are still many things we have to do to make the result closer to the global optimization and improve the performance of algorithm.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported in part by a Grant from The National Natural Science Foundation of China (61070122, 61070223, and 61373094); Jiangsu Provincial Natural Science Foundation (9KJA520002); Jiangsu Provincial Research Scheme of Natural Science for Higher Education Institutions (09KJA520002); Jiangsu Provincial Key Laboratory for Computer Information Processing Technology (kjs1024); and Jiangsu Province Support Software Engineering R&D Center for Modern Information Technology Application in Enterprise (SX200902).

References

[1] J. Shen, G. Gu, and H. Liu, "A survey of hierarchical reinforcement learning," *Pattern Recognition and Artificial Intelligence*, vol. 18, no. 5, pp. 574–581, 2005 (Chinese).

[2] C. Zonghai, W. Feng, N. Jianbin, and W. Xiaoshu, "A reinforcement learning method based on node-growing k-means clustering algorithm," *Journal of Computer Research and Development*, vol. 43, no. 4, pp. 661–666, 2006 (Chinese).

[3] J. J. F. Ribas-Fernandes, A. Solway, C. Diuk et al., "A neural signature of hierarchical reinforcement learning," *Neuron*, vol. 71, no. 2, pp. 370–379, 2011.

[4] H. Cuayáhuitl, S. Renals, O. Lemon, and H. Shimodaira, "Evaluation of a hierarchical reinforcement learning spoken dialogue system," *Computer Speech and Language*, vol. 24, no. 2, pp. 395–429, 2010.

[5] M. van Otterlo, "A survey of reinforcement learning in relational domains," Tech. Rep. TR-CTIT-05-31, University of Twente, 2005.

[6] S. Dzeroski, L. D. Raedt, and H. Blockeel, "Relational Reinforcement Learning," in *Advances in Proceedings ICML '98*, J. Shavlik, Ed., pp. 136–143, Morgan Kaufmann, Berlin, Germany, 2003.

[7] S. Sanner, "Simulteneous learning of structure and value in relational reinforcement learning," in *Advances in Proceeding of the ICML '05 Workshop on Rich Representations for Reinforcement Learning*, 2005.

[8] K. Driessens, J. Ramon, and T. Gärtner, "Graph kernels and Gaussian processes for relational reinforcement learning," *Machine Learning*, vol. 64, pp. 91–119, 2006.

[9] Q. Liu, Y. Gao, D. Chen, and Z. Cui, "A heuristic contour prolog list method used in logical reinforcement learning," *Journal of Information and Computational Science*, vol. 5, no. 5, pp. 2001–2007, 2008.

[10] Q. Liu, Y. Gao, Z. Cui, W. Yao, and Z. Chen, "An tableau automated theorem proving method using logical reinforcement learning," in *Advances in Computation and Intelligence*, vol. 4683 of *Lecture Notes in Artificial Intelligence*, Springer, 2007.

[11] R. S. Sutton, D. Precup, and S. P. Singh, "Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning," *Artificial Intelligence*, vol. 112, no. 1, pp. 181–211, 1999.

[12] T. G. Dietterich, "Hierarchical reinforcement learning with the MAXQ value function decomposition," *Journal of Artificial Intelligence Research*, vol. 13, pp. 227–303, 2000.

[13] R. Parr, *Hierarchical Control and Learning for Markov Decision Processes*, University of California, Berkeley, Calif, USA, 1998.

[14] E. G. Schultink, R. Cavallo, and D. C. Parkes, "Economic hierarchical Q-learning," in *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pp. 689–695, July 2008.

[15] S. Mannor, I. Menache, A. Hoze, and U. Klein, "Dynamic abstraction in reinforcement learning via clustering," in *Proceedings of the 21st International Conference on Machine Learning*, pp. 560–567, ACM Press, Banff, Canada, July 2004.

[16] M. Stolle and D. Precup, "Learning options in reinforcement learning," in *Proceedings of the 5th International Symposium on Abstraction, Reformulation and Approximation*, pp. 212–285, Kananaskis, Canada, 2002.

[17] Ö. Şimşek, A. P. Wolfe, and A. G. Barto, "Identifying useful subgoals in reinforcement learning by local graph partitioning," in *Proceedings of the 22nd International Conference on Machine Learning*, pp. 248–256, ACM Press, August 2005.

[18] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, Mass, USA, 1998.

[19] G. Yang, C. Shifu, and L. Xin, "Research on reinforcement learning technology: a review," *Acta Automatica Sinica*, vol. 33, no. 1, pp. 86–99, 2004.