Using Control Theory for Analysis of Reinforcement Learning and Optimal Policy Properties in Grid-World Problems

S. Mostapha Kalami Heris[1], Mohammad-Bagher Naghibi Sistani[2], and Naser Pariz[2]

[1] Control Engineering Department, Faculty of Electrical Engineering,
K. N. Toosi University of Technology, Tehran, Iran
[2] Electrical Engineering Department, Faculty of Engineering,
Ferdowsi University of Mashhad, Mashhad, Iran
kalami@ee.kntu.ac.ir, mb-naghibi@um.ac.ir, n-pariz@um.ac.ir

Abstract. Markov Decision Process (MDP) has enormous applications in science, engineering, economics and management. Most of decision processes have Markov property and can be modeled as MDP. Reinforcement Learning (RL) is an approach to deal with MDPs. RL methods are based on Dynamic Programming (DP) algorithms, such as Policy Evaluation, Policy Iteration and Value Iteration. In this paper, policy evaluation algorithm is represented in the form of a discrete-time dynamical system. Hence, using Discrete-Time Control methods, behavior of agent and properties of various policies, can be analyzed. The general case of grid-world problems is addressed, and some important results are obtained for this type of problems as a theorem. For example, equivalent system of an optimal policy for a grid-world problem is dead-beat.

Keywords: Dynamic Programming, Discrete-Time Control Systems, Markov Decision Process, Reinforcement Learning, Stochastic Control.

1 Introduction

A grid-world problem is a special case of Markov Decision Process (MDP). MDP is introduced and studied by Bellman (1957) for the first time, and studied in more detail by Howard (1960) [1]. First registered application of MDP is Road Management in Arizona State (1978) [1]. Wild-life Management, Manufacturing Management, Inventory and Transportation, are examples of MDP applications [1]. A full list of MDP applications is gathered by Puterman [1], Cassandra [2], Pyeatt [3], Hu et al. [18] and Soo et al. [19].

Obtaining an optimal policy for a MDP is one of most discussable problems in optimization and control theories. To solve this problem, Linear Programming (LP) and Dynamic Programming (DP) methods are used [1,5,8,18,19]. Basic DP algorithms to deal with MDPs, are Policy Evaluation, Policy Iteration and Value Iteration, which are studied in [1], [4], [5], [6] and [7], in detail. Use of these methods, yields accurate solutions to MDPs, and also inexact solutions with specific accuracy can be obtained. Existence of solutions for these methods, is proved using various methods and Exact

D.-S. Huang et al. (Eds.): ICIC 2009, LNAI 5755, pp. 276–285, 2009.

conditions for the existence of optimal solution for MDP, are gathered and studied in detail [5,6,7,16]. Curse of Dimensionality, which is introduced by Bellman, causes critical problems with the use of DP algorithms within large-scale problems. So approximate and fast algorithms are developed in the framework of Reinforcement Learning (RL) [5,7,9] and Neural-Dynamic Programming [4,8,9].

The approaches deployed for improvement of the procedure used to solve MDP are dividable into four major groups. Methods of group one use structural properties of the problem to facilitate procedure of achieving solution or optimal policy [11,12,26]. Other methods in opposition to methods discussed in group one, do not lead to optimal results for the problem. Hence these results are sub-optimal [10,15]. Approaches in group two could be divided into two sub categories. In methods of first sub-group, simplified models are used to solve the problem and sub-optimal solutions are found for main problem [10,15]. In the second sub-group methods, the structure of the policy is being assumed in a particular form and therefore a kind of simplification is implemented [4,6,10,15]. The third group of approaches is obtained through approximation of State Value Functions and State-Action Value Function, and equations of DP [4,5,7,9,10,13,14]. Methods in the fourth group are defined and used in policy space such as classic method of Policy Iteration. These approaches are aimed to speed up formation of optimal policy using special methods and consequently the overall process of problem solution is accelerated.

One of suggested methods in order to speed up the process of solution of MDP, is utilization of a method called Policy Reuse, which is used to accelerate the learning and categorization of information obtained on later experiences [20-23]. Also optimization of policy iteration algorithm has resulted in appropriate consequences. We can refer to [1], [6], [7], [19], [24], [25] and [26] as examples.

Many approaches are introduced to improve functionality and speed of DP algorithms and also RL algorithms in MDPs. The main difficulty in most of problems is absence of knowledge about optimal result. There is no simple and fast criterion to compare two policies and to identify an optimal policy. It means that the policy must be utilized by the agent to evaluate it, based on the return. In spite of many researches carried on to improve functionality of algorithms related to solution of MDP problems, no method is proposed to mathematical analysis of optimal policy in MDP problems, up to now. For this reason there is no criterion to analyze performance of the agent, which obeys particular policy, but time consuming algorithms.

In this paper the procedure of solving MDP problem using Dynamic Programming method is stated as a discrete-time or digital dynamical system. Obtained digital system is consistent with proposed solution for this problem. It is possible to identify related solution properties by analyzing control characteristics of the equivalent system and guessing quality of final solution. Utilizing this equalization leads to the possibility of analyzing functionality of RL in Markovian environments.

Other sections of this paper are as follows. In the section 2, Reinforcement Learning and its main ideas are discussed briefly. Section 3 studies primary definitions of MDP and DP methods, which are used to solve this type of problems, are represented. In section 4, policy evaluation algorithm is proposed as a discrete-time dynamical system. In section 5, two sample problems are investigated based on contents of section 4 and analysis and control methods of discrete-time dynamical

systems. Section 6 consists of general statements on grid-world problems and optimal policy related to this type of problems. Finally, conclusion is done in section 7.

2 Reinforcement Learning

In Reinforcement Learning the main purpose of learning is performing duties or reaching some target without the case that the agent is fed by direct external information. In this method the only way of informing agent is through one reward or penalty signal. The one thing that gets across to agent via reward signal is whether appropriate decision is made or not. In this method of learning the agent is never told that which the correct action in each state is, and just it is being told that how much each action is good or bad. A generic Reinforcement Learning problem is shown in Fig. 1. Learning agent achieves some descriptions of environment through sensors. Information relevant to the environment is fed to the agent as sensory information. When the agent performs an action, it earns a reward which could be positive or negative based on whether the action was good or bad. The target of the agent in learning is equivalent to maximization of reward in a specific range of time. These two targets are consistent and satisfaction of each one will result in satisfaction of the other. In this way the agent learns appropriate way of operation via focus on earned rewards. This is possible through finding a mapping between states and actions doable by the agent. This mapping which is called policy tells the agent what to perform when run into different states.

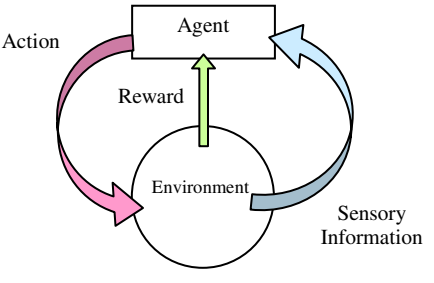

Fig. 1. One Reinforcement Learning problem and the way agent and environment interact

3 Markov Decision Processes

One major assumption in most of works on Reinforcement Learning is that the interaction between agent and surrounding environment could be modeled as a Markov Decision Process (MDP). A MDP is a stochastic and discrete-time process which is often defined as quadruple like $(\mathbb{S}, \mathbb{A}, \mathcal{P}, \mathcal{R})$.

$\mathbb{S} = \{s^1, s^2, \ldots, s^n\}$ is state space of the process and consists of all possible states that deciding agent encounter and the agent must make a decision. $\mathbb{A} = \{a^1, a^2, \ldots, a^m\}$ is set of actions which could be selected by the agent. $\mathcal{P}$ models transition and change

of states. If the process is in state s and action a is selected by the agent, the probability that the state of process change to s' is defined as $\mathcal{P}_{ss'}^{a}$. Generally speaking, $\mathcal{P}$ is a mapping of the form $\mathcal{P}:\mathbb{S}\times\mathbb{A}\times\mathbb{S}\to[0,1]$. This model has the Markov property. If s_t and a_t indicate process state and selected action in discrete time t respectively, then Markov property could be defined as below:

$$\mathcal{P}_{s_t s_{t+1}}^{a_t} = \Pr\{s_{t+1}\,|\,s_t, a_t\} = \Pr\{s_{t+1}\,|\,s_t, a_t, s_{t-1}, a_{t-1}, \ldots\} \ . \tag{1}$$

$\mathcal{R}$ defines expectation of reward. If the agent is in state s and goes to state s' by choosing action a, the value of reward will be a random number with expectation of $\mathcal{R}_{ss'}^{a}$. Generally speaking, reward function is a mapping such as $\mathcal{R}:\mathbb{S}\times\mathbb{A}\times\mathbb{S}\to\mathbb{R}$. Function $\mathcal{R}$ has the Markov property and the value of $\mathcal{R}_{ss'}^{a}$ is just dependent to present state s and present action a and next state s' and is totally independent of previous states or actions.

The probability that action a is selected by the agent in state s is being defined as a mapping such as $\pi:\mathbb{S}\times\mathbb{A}\to[0,1]$, such that $\Pr\{a_t = a\,|\,s_t = s\} = \pi(s,a)$. The mapping π is known as policy and is the main unknown of a Reinforcement Learning problems or every other decision making problem [1,5,7,14].

In order to compare different policies with each other, it is possible to define a criterion to measure them. This criterion is the value that each policy returns in each state of process and is called as policy return in mentioned state. The return of one policy is the value that is obtained during consecutive decisions and obeying that policy. Solving a Reinforcement Learning problem is to find optimal policy of π^* in the way that maximizes the return of policy or the value of each state [5,7,14]. If a constant discount factor is defined as $\gamma\in[0,1]$, then the value of state s could be calculated by:

$$V^{\pi}(s) = \mathbb{E}^{\pi}\{r_{t+1} + \gamma r_{t+2} + \cdots\,|\,s_t = s\} = \sum_{a\in\mathbb{A}}\pi(s,a)\sum_{s'\in\mathbb{S}}\mathcal{P}_{ss'}^{a}\left(\mathcal{R}_{ss'}^{a} + \gamma V^{\pi}(s')\right) \ , \tag{2}$$

in which $\mathbb{E}^{\pi}$ is the operator of expectation. Super-indices π are written to emphasize that the agent obeys policy π. Equation (2) is known as Bellman optimality equation [1,4,14,18,19]. One of the methods being used to solve this equation and finding values of all states is to change the equations into recurrent mode. This method, which is based on fixed-point theorem [17], and it is suggested to rewrite the equation (2) as following form[17]:

$$V_{k+1}^{\pi}(s) = \sum_{a\in\mathbb{A}}\pi(s,a)\sum_{s'\in\mathbb{S}}\mathcal{P}_{ss'}^{a}\left(\mathcal{R}_{ss'}^{a} + \gamma V_{k}^{\pi}(s')\right) \ , \tag{3}$$

in which $V_k^{\pi}(s)$ is k^{th} estimation of actual value of $V^{\pi}(s)$. Based on the fact that $|\gamma|<1$, we can infer $V_{k+1}^{\pi}(s)$ is related to $V_k^{\pi}(s)$ through a contraction mapping [17] and this mapping has a unique fixed-point which is the solution of equation and:

$$V^{\pi}(s) = \lim_{k\to\infty} V_k^{\pi}(s) \ . \tag{4}$$

4 Modeling the Policy Evaluation algorithm

Assume a vector defined as follows:

$$v^{\pi} = [V^{\pi}(s^1) \quad V^{\pi}(s^2) \quad \cdots \quad V^{\pi}(s^n)]^T \ . \tag{5}$$

This vector consists of all values for states of a model. Therefore recurrent equation of (3) could be rewritten for all states and changed to a joint equation:

$$v_{k+1}^{\pi} = \gamma \mathcal{P} v_k^{\pi} + \mathcal{R} = \gamma \begin{bmatrix} \mathcal{P}_{s^1 s^1} & \cdots & \mathcal{P}_{s^1 s^n} \\ \vdots & \ddots & \vdots \\ \mathcal{P}_{s^n s^1} & \cdots & \mathcal{P}_{s^n s^n} \end{bmatrix} v_k^{\pi} + \begin{bmatrix} \mathcal{R}_{s^1} \\ \vdots \\ \mathcal{R}_{s^n} \end{bmatrix}, \tag{6}$$

in which elements of matrices $\mathcal{P}$ and $\mathcal{R}$ are defined as follows:

$$\mathcal{P}_{ss'} = \sum_{a \in \mathbb{A}} \pi(s,a) \mathcal{P}_{ss'}^{a} \ , \text{ and} \tag{7}$$

$$\mathcal{R}_s = \sum_{a \in \mathbb{A}} \sum_{s' \in \mathbb{S}} \pi(s,a) \mathcal{P}_{ss'}^{a} \mathcal{R}_{ss'}^{a} = \sum_{a \in \mathbb{A}} \pi(s,a) \mathcal{R}_s^{a} \ . \tag{8}$$

Equation (6) could be rewritten in the form of $v_{k+1}^{\pi} = \gamma \mathcal{P} v_k^{\pi} + \mathcal{R} u_k$, in which it is assumed that u_k is discrete-time unit step, which is equals to 1 for $k \geq 0$ [27]. Equation $v_{k+1}^{\pi} = \gamma \mathcal{P} v_k^{\pi} + \mathcal{R} u_k$, is state equation of a discrete-time linear system [27], which its state variables are values of the process states. Stability condition for this system is that, all eigenvalues of matrix $\gamma \mathcal{P}$, are located inside unit circle [27]. To satisfy this condition, discount factor must agree the following inequality:

$$\gamma < \frac{1}{\max\limits_{1 \leq i \leq n} |\lambda_i(\mathcal{P})|} = \frac{1}{\rho(\mathcal{P})} \ , \tag{9}$$

in which $\lambda_i(\mathcal{P})$ is the i^{th} eigenvalue and $\rho(\mathcal{P})$ is spectral radius of square matrix $\mathcal{P}$. It can be seen that the condition $\gamma \leq 1$ does not necessarily ensure existence of finite solution for (2). Equation (9) states more accurate condition for γ .

5 Solving and Investigating a Problem

Assume a table such as Fig. 2. In this problem the agent is located in one of white cells again and it is required to reach one of gray cells moving up, down, left and right. Moving in each direction has a reward of -1 which shows the cost to move in each state. The movements which cause agent exit the table has no effect on the location of the agent. The agent must learn to reach one of target cells earning maximum reward (minimum penalty). If so the agent has reach target in minimum number of movements. This problem could be stated as a MDP.

s^0	s^1	s^2	s^3
s^4	s^5	s^6	s^7
s^8	s^9	s^{10}	s^{11}
s^{12}	s^{13}	s^{14}	s^0

Fig. 2. The table relevant to the problem in section 5.2

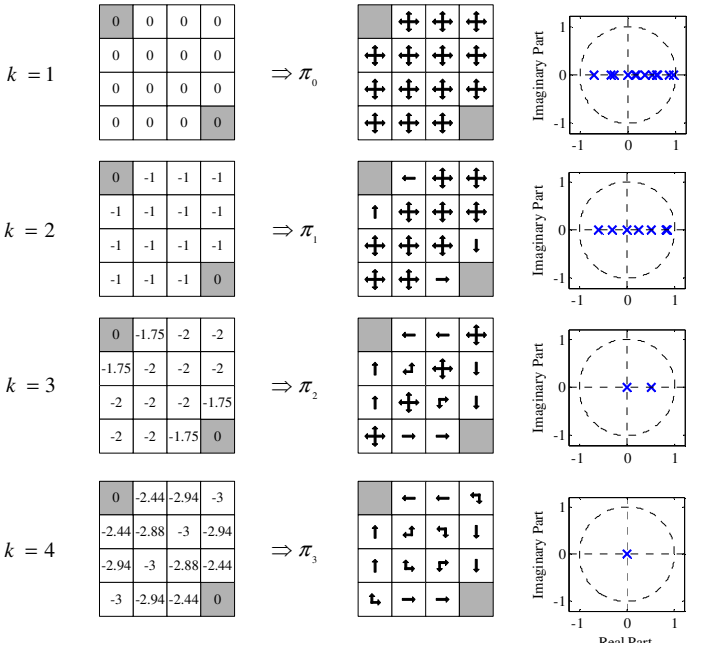

Fig. 3. Steps of policy evaluation algorithm for π_0 and inferred policies in each step with the diagram of equivalent dynamic system poles

In order to deploy policy evaluation method, initial value of all cells are assumed to be zero. The policy deployed up to the end of evaluation, is random policy. It means that in all cells of the table the probability of movements in all directions are the same and all equal to 0.25. In Fig. 3 some iterations of Eq. (3) are shown.

Based on the results of each step, a greedy policy could be formed. It means that the agent must move in the direction which has the most value. The policy obtained using the results yielded in step k^{th} is shown as π_k . π_0 is the random policy. π_3 and all of policies after that, are identical and this fact is referred to by $\pi_3 = \pi_\infty$. Assume matrix $\mathcal{P}$ in equation (6) equals $\mathcal{P}_i$ for policy π_i . Spectral radius of all mentioned matrixes are calculated and shown as follows:

$$\rho(\mathcal{P}_0) \simeq 0.9468, \quad \rho(\mathcal{P}_1) \simeq 0.8431, \quad \rho(\mathcal{P}_2) = 0.5, \quad \rho(\mathcal{P}_3) = \rho(\mathcal{P}_\infty) = 0 \tag{10}$$

It can obviously be seen that more optimal the policy, the least the greatest eigenvalue of state matrix is. Specially for policy π_3, all eigenvalues are zero. Consequently upper bound for discount coefficient value of γ are as follows respectively: 1.0562, 1.1861, 2, ∞. So the Eq. (2) has finite solutions for all values of γ, for the policy π_3.

The poles of system (6) are identical to eigenvalues of matrix $\gamma\mathcal{P}$. It can be seen that the poles of the system equal to policy π_∞ are all located at the origin, the system is dead-beat and has the fastest response among the systems with the same degree. Defining the value of all states as the output, transfer function of all systems could be found as $G_i(z) = (zI - \gamma\mathcal{P}_i)^{-1}\mathcal{R}_i$. This transfer function models a system with one input and 14 outputs. Each output is related to the value of one cell in the grid. Frequency response of four systems are shown in Fig. 4 for cell s^3.

The scaling of vertical axis is not set to dB and mentioned values are real values. The frequency response shown in Fig. 4 contains very important information about the environment and the policy implemented by the agent. In this figure, steady state response of the systems in cell s^3 can be observed. Steady state response of these systems is identical to the value of frequency response in frequency zero. It can be seen that the amplitude of frequency response in frequency zero for policies π_0, π_1, π_2 and π_3 is respectively 22, 9, 4 and 3. These are mean costs that agent pays to reach each of target cells starting from s^3. According to the definition of reward for this problem the value of penalty is the number of movements which the agent does to reach target cells. It can be seen that the minimum number of movements is achieved for π_3. Fig. 4 shows that the systems equivalent to better policies have wider bandwidth. Based on the fact that the bandwidth is a criterion of response speed, it can be inferred that for better policies response speed of systems is faster. This is the point we expect from a good policy for present problem.

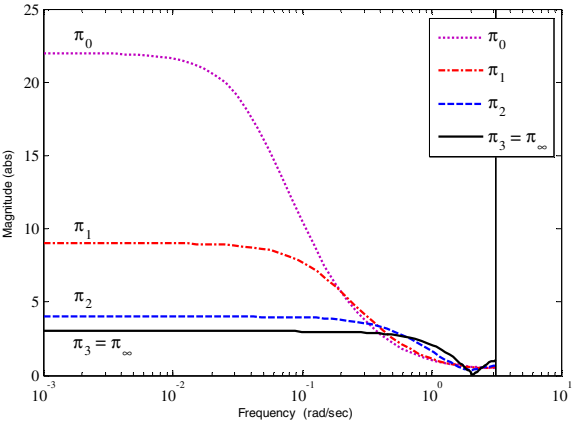

Fig. 4. Frequency response of systems obeying policies π_0 to π_3 in cell s^3 of table in Fig. 2

6 General Investigation of Grid-World Problems

The results obtained in this section are related to grid-world problem or other similar problems. Common characteristics of the problems for which the results of this section are applicable, are listed:

- The problem must be representable in the form of MDP.
- There must be one or more final state that reaching them means end of process.
- All movements have cost as negative reward.
- The target of the problem is to find a schedule to reach final state paying minimum cost or yielding minimum negative reward. Total cost is defined as discounted summation of rewards obtained by the agent.

Assume a grid-world problem with one or more target cells. All target cells are assumed as one joint state. The value of state relevant to target cells is always assumed to be zero and dismissed from the vector of state values. Each movement has a cost modeled as a -1 reward. For each state s, at least one path could definitely be found in a way that has the least penalty. This path has the least possible movements and for the state s this number of movements is indicated by $m(s)$. Under these assumptions, it is possible to state a theorem about grid-world and similar problems.

Theorem. Dynamical system equal to optimal policy evaluation system of a table problem has the properties as follows:

- The system is a dead-beat one and the component relevant to state s has exactly $m(s)$ poles at the origin of z plane.
- Frequency response of this system in frequency zero has the least value. If $\gamma = 1$, the magnitude of frequency relevant to state s in $\omega = 0$ will be equal to $m(s)$. This value equals to the final value of state s with negative sign.
- This system is stable for all values of discount factor γ. □

Proof. If the agent obeys optimal policy, it will pass an optimal path starting from state s and will reach target cell exactly after doing $m(s)$ movements and finally will have a total reward equal to $-m(s)$. If we assume $\gamma = 1$ the value of cell s after movement k^{th} will be summation of rewards gained by the agent up to k^{th} movement. Therefore the value of state s could be defined as a function of discrete time as:

$$V_k(s) = \begin{cases} 0 & ,k < 0 \\ -k & ,0 \le k \le m(s) \\ -m(s) & ,k > m(s) \end{cases} . \tag{11}$$

Note that the time in definition above is equal to the number of iteration in policy evaluation algorithm. The function above is one of outputs for equivalent system to optimal policy evaluation. z transform of mentioned value function is as follows:

$$\mathcal{Z}\{V_k(s)\} = V_s(z) = -\sum_{k=1}^{m(s)} k z^{-k} - m(s) \sum_{k=m(s)+1}^{\infty} z^{-k} . \tag{12}$$

At the other hand, based on the proposed contents, the input of the system equal to policy evaluation is always assumed to be unit step. Therefore the transfer function relevant to state s could be calculated through following equation:

$$V_s(z) = G_s(z)U(z) = \frac{1}{1-z^{-1}}G_s(z) \ . \tag{13}$$

Therefore the transfer function relevant to value of state s , is defined by $G_s(z) = (1-z^{-1})V_s(z)$. Substituting (12) yields:

$$G_s(z) = -\sum_{k=1}^{m(s)} z^{-k} = -\frac{z^{m(s)-1}+\cdots+z+1}{z^{m(s)}} = -\frac{z^{m(s)}-1}{z^{m(s)}(z-1)} \ . \tag{14}$$

It can be seen that the transfer function relevant to state s has exactly $m(s)$ poles on the origin of z plane, and $G_s(z)$ is a dead-beat system. Also zeros of $G_s(z)$ are always located on the unit circle and all of them are $m(s)^{\text{th}}$ roots of 1. Note that $z=1$ is not a zero of $G_s(z)$. ■

7 Conclusion

In this paper having a survey on Reinforcement Learning (RL), Markov Decision Process (MDP) and Dynamic Programming (DP), equations related to solution of one MDP using DP are rewritten and concluded as a discrete-time dynamical system. This approach provides the possibility of solving and analyzing a Reinforcement Learning problem in Markovian environment in the form of a discrete-time dynamical system. Therefore it is possible to use Discrete-Time Control methods to analyze a learning process. The main subject of this paper is on a kind of problems called grid-world problems. The results show that the optimal policies for this type of problem could be defined as dead-beat system in the framework of Discrete-Time Control systems. Generalization of these results into other problems and defining a duality between decision making space and Discrete-Time Control system space are of completing researches and studies could be imagined.

References

1. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley, Chichester (2005)
2. Cassandra, A.: Exact and Approximate Algorithms for Partially Observable Markov Decision Processes. Ph.D. Thesis, Brown University (1998)
3. Pyeatt, L.: Integration of Partially Observable Markov Decision Processes and Reinforcement Learning for Simulated Robot Navigation. Ph.D. Thesis, Colorado State University (1999)
4. Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific (1996)
5. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. The MIT Press, Cambridge (1998)

6. Lew, A., Mauch, H.: Dynamic Programming: A Computational Tool. Springer, Berlin (2007)
7. Reynolds, S.I.: Reinforcement Learning with Exploration. Ph.D. Thesis, School of Computer Science, University of Birmingham, UK (2002)
8. Van Roy, B.: Neuro-Dynamic Programming: Overview and Recent Trends. In: Feinberg, E.A., Schwartz, A. (eds.) Handbook of Markov Decision Processes: Methods and Applications. Kluwer Academic, Dordrecht (2002)
9. Si, J., et al.: Handbook of Learning and Approximate Dynamic Programming. Wiley InterScience, Hoboken (2004)
10. Soo Chang, H., et al.: A survey of some Simulation-Based Algorithms for Markov Decision Processes. Communications in Information and Systems 7(1), 59–92 (2007)
11. Smith, J.E., Mc Cardle, K.F.: Structural Properties of Stochastic Dynamic Programs. Operations Research 50, 796–809 (2002)
12. Fu, M.C., et al.: Monotone optimal policies for queuing staffing problem. Operations Research 46, 327–331 (2000)
13. Givan, R., et al.: Bounded Markov Decision Processes. Artificial Intelligence 122, 71–109 (2000)
14. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research 4, 237–285 (1996)
15. Gordon, G.J.: Approximate Solution to Markov Decision Processes. Ph.D. Thesis, School of Computer Science, Carnegie Mellon University (1999)
16. de Farias, D.P., Van Roy, B.: On the Existance of Fixed-points for Approximate Value Iteration and Temporal-Difference Learning. Journal of Optimization theory and Applications 105(3), 589–608 (2000)
17. Royden, H.: Real Analysis, 3rd edn. Prentice Hall, Englewood Cliffs (1988)
18. Hu, Q., Yue, W.: Markov Decision Processes with Their Applications. Springer Science+Busines Media, LLC (2008)
19. Soo, H., et al.: Simulation-based Algorithms for Markov Decision Processes. Springer, London (2007)
20. Fernandez, F., Veloso, M.: Exploration and Policy Reuse. Technical Report, School of Computer Science, Carnegie Mellon University (2005)
21. Fernandez, F., Veloso, M.: Probabilistic Reuse of Past policies. Technical Report, School of Computer Science, Carnegie Mellon University (2005)
22. Fernandez, F., Veloso, M.: Building a Library of Policies through Policy Reuse. Technical Report, School of Computer Science, Carnegie Mellon University (2005)
23. Bernstein, D.S.: Reusing Old Policies to Accelerate Learning on New Markov Decision Processes. Technical Report, University of Massachusetts (1999)
24. Zhang, N.L., Zhang, W.: Speeding Up the Convergence of Value Iteration in Partially Observable Markov Decision Processes. Journal of Artificial Intelligence Research 14, 29–51 (2001)
25. Hansen, E.A.: An Improved Policy Iteration for Partially Observable Markov Decision Processes. In: Proceedings of 10th Neural Information Processing Systems Conference (1997)
26. Sallans, B.: Reinforcement Learning for Factored Markov Decision Processes. Ph.D. Thesis, Graduate Department of Computer Science, University of Toronto (2002)
27. Ogata, K.: Discrete-Time Control Systems, 2nd edn. Prentice Hall, Englewood Cliffs (1994)