INTELLIGENT AGENTS APPROACH TO THE ON-LINE MANAGEMENT AND CONTROL OF TRANSPORTATION NETWORKS

Nagi Basha[1] and Adel Sadek[2]

ABSTRACT

The primary objective of Intelligent Transportation Systems (ITS) is to take advantage of the advances in communications, electronics, and Information Technology in order to improve the efficiency and safety of transportation systems. Among the several ITS applications is the notion of Dynamic Traffic Routing (DTR), which involves generating optimal routing recommendations to drivers with the aim of maximizing network utilizing. The objective of this study is to demonstrate the feasibility of using a self-learning intelligent agent to solve the DTR problem to achieve traffic user equilibrium in a transportation network. To do this, the study uses the Cell Transmission Model to simulate a simple network that has two main routes. The learning algorithm implemented in the agent is based on the SARSA algorithm which is one variation of Reinforcement Learning (RL). SARSA allows the agent to learn by itself through interaction with the simulation model. Preliminary results indicate that the agent is capable of learning the correct strategies for the different states of the problem, provided that each state is visited long enough. Once the agent reaches a certain degree of expertise, it can be deployed to a real transportation system where the agent can use his knowledge and learned expertise to provide online routing guidance for motorists. Moreover, since any real life transportation system is a stochastic and ever-changing system, the agent will continue refining and adapting its expertise through its communication with the real transportation system.

KEY WORDS

Dynamic Traffic Routing, Intelligent Transportation Systems, Reinforcement Learning, Artificial Intelligence, Traffic Simulation

[1] Ph.D. Candidate, Dept. of Computer Science, University of Vermont, Burlington, VT, 05405, USA, , nbasha@uvm.edu

[2] Associate Professor, Depart. of Civil and Environmental Engineering and Department of Computer Science, University of Vermont, Burlington, VT, 05405, USA, Phone 802/656-4126, Fax:(802) 656-8446, asadek@cems.uvm.edu

INTRODUCTION

In recent years, there has been a concerted effort aimed at taking advantage of the advances in communications, electronics, and Information Technology in order to improve the efficiency and safety of transportation systems. Within the transportation community, this effort is generally referred to as the Intelligent Transportation Systems (ITS) program. From an efficiency standpoint, ITS offers an alternative approach to meeting the challenges of increasing travel demand that does not require the physical construction of additional transportation system capacity, but rather an approach that attempts to optimize the utilization of existing capacity. Among the primary ITS applications is the notion of Dynamic Traffic Routing (DTR), which involves routing traffic in real-time so as to maximize the utilization of existing capacity. The solution to the DTR problem involves determining the *time-varying* traffic splits at the different diversion points of the transportation network. These splits could then be communicated to drivers via Dynamic Message Signs or In-vehicle display devices.

Solving the DTR problem is quite challenging, and existing approaches to solving the DTR problem have their limitations. This paper proposes a solution for highway dynamic traffic routing based on a self-learning intelligent agent. The core idea is to deploy an agent to a simulation model of a highway. The agent will then learn by itself through interacting with the simulation model. Once the agent reaches a satisfactory level of performance, it could then be deployed to the real world, where it would continue to learn how to refine its control policies over time. The advantages of such approach are quite obvious given the fact that real-world transportation systems are stochastic and ever-changing, and hence are in need of on-line, adaptive agents for their management and control.

REINFORCEMENT LEARNING

Among the different paradigms of soft computing and intelligent agents, Reinforcement Learning (RL) appears to be particularly suited to address a number of the challenges of the on-line DTR problem. RL involves learning what to do and how to map situations to actions to maximize a numerical reward signal (Kaelbling, 1996; Kretchmar, 2000; Abdulhai and Kattan, 2003; Russell and Norvig, 2003). A Reinforcement Learner Agent (RLA) must discover on its own which actions to take to get the most reward. The RLA will learn this by trial and error. The agent will learn from its mistakes and come up with a policy based on its experience to maximize the attained reward. Figure 1 depicts a typical RLA and its relationship with the environment (Sutton and Barto, 2000).

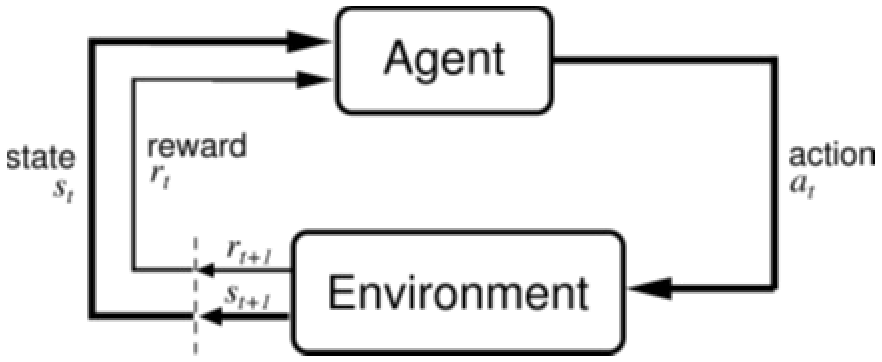

Figure 1: The agent-environment interaction in reinforcement learning

The field of applying RL to transportation management and control applications is still in its infancy. A very small number could be identified from the literature. A Q-learning algorithm (which is a specific implementation of reinforcement learning) is introduced in Abdulhai et. al. (2003) to study the effect of deploying a learning agent using Q-learning to control an isolated traffic signal in real-time on a two dimensional road network. The performance of the Q-learning agent is tested under different traffic circumstances to see how it will manage the traffic under varying conditions. The Q-learning outperformed the pre-timed signal in the case of constant-ratio flow rates. However, in the case of uniform flow rates, the Q-learning agent's results were not statistically significant. Abdulhai and Pringle (2003) extended this work to study the application of Q-learning in a multi-agent context to manage a linear system of traffic signals. The advantage of having a multi-agent control system is to achieve robustness by distributing the control rather than centralize it even in the event of communication problems. Finally, Choy et al. (2003) develop an intelligent agent architecture for coordinated signal control and use RL to cope with the changing dynamics of the complex traffic processes within the network.

PURPOSE AND SCOPE

The main purpose of this study is to show the feasibility of using RL for solving the problem of providing online Dynamic Route Guidance for motorists, through providing a set of experiments that show how an RL-based agent can provide reasonable guidance for a simple network that has two main routes. Figure 2 shows the simple network used in this study. It should be noted that this network is largely similar to the test network used by Wang et al. (2003) in evaluating predictive feedback control strategies for freeway network traffic.

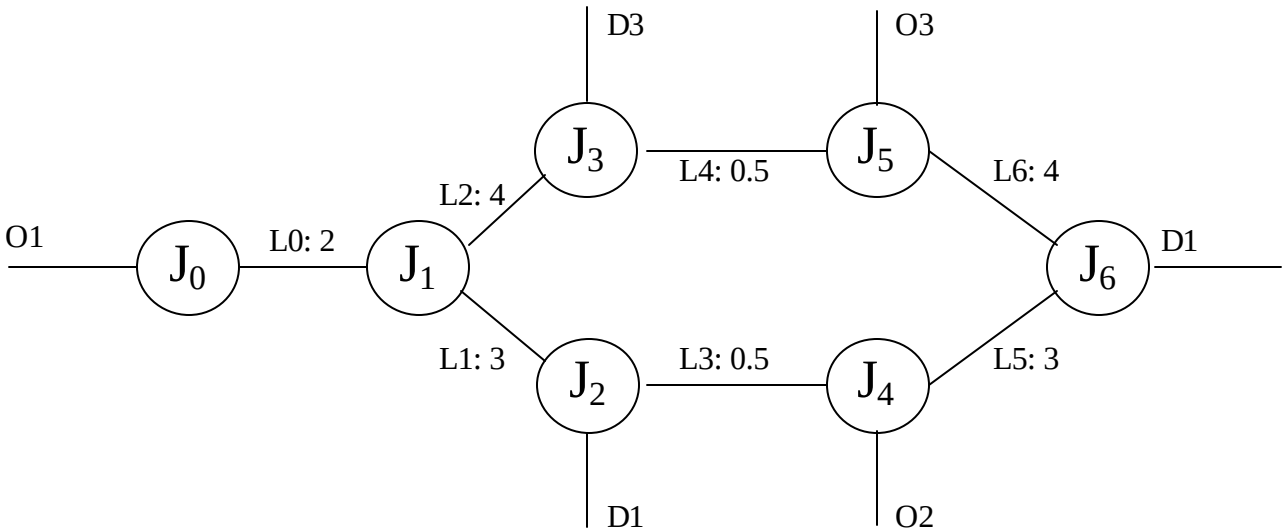

Figure 2: Network Topology

The network has three origins: O1, O2, and O3 and three destinations D1, D2, and D3. Each origin generates a steady flow of traffic. Traffic disappears when it reaches any of the three destinations. The length in miles of each link is indicated on the graph. For example, L0 has a length of 2 miles. The capacity of all links is 4000 veh/h except for L0 that has a capacity of 8000 veh/h. All links have two lanes except L0 that has 4 lanes.

As can be seen from Figure 2, there are two alternate routes connecting origin O1 to destination D1. The primary route (route A) goes through nodes J1, J2, J4 and J6, and has a total length of 6.50 miles. The second route (route B) goes through nodes J1, J3, J4 and J6. Route B, with a total length of 8.50 miles, is therefore longer than route A. The intelligent RL agent is deployed right at the J1 junction. The goal of the agent is to determine an appropriate diversion rate at J1 so as to achieve traffic user equilibrium between the two routes connecting zones O1 and D1 (i.e. so that travel times along routes A and B are as close to each other as possible), taking into consideration the current state of the system. For example, if there is major congestion at L1, a well-experienced agent should advice motorists to divert to L2 because by doing that, eventually the congestion will be cleared. To simplify the case study, the diversion rates at J2, and J3 are set statically in the model; i.e. they are fixed numbers throughout the length of the simulation experiment.

METHODOLOGY

CELL TRANSMISSION MODEL

The first task was to build a simulation model for the test network shown in Figure 2. In this study, we selected the Cell Transmission Model (CTM) to build the simulation model, with

which the RL agent would interact to learn for itself the best routing strategies. The CTM was developed by Daganzo to provide a simple representation of traffic flow capable of capturing transient phenomena such as the propagation and dissipation of queues (Daganzo, 1994; 1995). The model is macroscopic in nature, and works by dividing each link of the roadway network into smaller, discrete, homogeneous cells. Each cell is appropriately sized to permit a simulated vehicle to transverse the cell in a single time step at free flow traffic conditions. The state of the system at time t is given by the number of vehicles contained in each cell, $n_i\ (t)$. If cells are numbered consecutively from the upstream end of the roadway from $i = 1$ to I, the recursive relationship of the cell-transmission model can be written as:

$$n_i\ (t + 1) = n_i\ (t) + y_i\ (t) - y_{i+1}\ (t) \qquad \text{[Equation 1]}$$

where $y_i\ (t)$ is the inflow to cell i in the time interval $(t, t+1)$ given by:

$$y_i\ (t) = \min\ \{n_{i-1}\ (t), Q_i(t), d[N_i\ (t) - n_i\ (t)]\} \qquad \text{[Equation 2]}$$

where $Q_i\ (t)$ is the maximum number of vehicles that can flow into cell i in the time interval $(t, t+1)$, $N_i\ (t)$ is the maximum number of vehicles that can be present in cell i at time t, and d is the ratio of the shock wave speed (w) to the free flow speed (v). Daganzo showed that if the relationship between flow (q) and density (k) is of the form shown in Figure 3, then the cell-transmission model can be used to approximate the kinematic wave model of Lighthill and Whitham (1955).

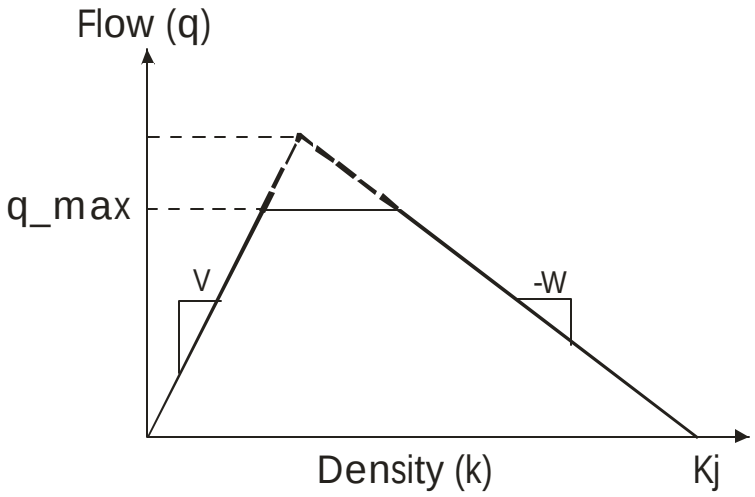

Figure 2: Flow Density Relationship

As can be seen from Figure 3, the cell transmission model offers the user four degrees of freedom (i.e. four parameters that need to be specified). These are: (1) the free flow speed (v) which determines the length of the cells or the time step of the cell-transmission model; (2) q_{max} which determines the maximum number of vehicles that can flow into cell i, $Q_i(t)$; (3) the shock wave speed which determines the parameter, d, of Equation 2; and (4) the jam density, k_j, which determines the maximum number of vehicles that can be present in cell i, $N_i(t)$. By adjusting these four parameters, the user can calibrate the model and bring its

results closer to reality. For this study, a C++ implementation of the CTM was developed and used to simulate the test network.

THE INTELLIGENT AGENT

As previously mentioned, RL was the paradigm chosen to develop the intelligent, learning agent that will be used for dynamic traffic routing. Specifically, the learning algorithm implemented in the agent is based on the SARSA algorithm, which is an on-policy Temporal Difference (TD) learning implementation of reinforcement learning (Sutton and Barto, 2000). It is called on-policy because the agent keeps on exploring and refining the policy that it is currently using. This means that the agent enhances and modifies the policy it follows to make decisions. This is different from off-policy learning techniques, where the agent follows one policy and enhances another policy that could be totally different than the one the agent uses to make decisions.

SARSA is a temporal difference algorithm because – like Monte Carlo ideas - it can learn directly from the experience without requiring a model of the dynamics of the environment. Like Dynamic Programming ideas (Bertsekas, 2000), SARSA updates the desirability of its estimates of state-action pairs, based on earlier estimates; i.e. SARSA does bootstrapping. For a complex, unpredictable, and stochastic system like a transportation system, SARSA seemed to be very suitable to adapt with the nature of an ever-changing system.

The implementation of the SARSA algorithm is quite simple. Each state action pair (s,a) is assigned an estimate of the desirability of being in state s and doing action a. The desirability of each state action pair can be represented by a function Q(s,a). The idea of SARSA is to keep updating the estimates of Q(s,a) based on earlier estimates of Q(s,a) for all possible states and all possible actions that can be taken in every single state. Equations 3 shows how the Q(s,a) values are updated.

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha\,[r_{t+1} + Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)] \qquad \text{[Equation 3]}$$

where α is the step-size parameter or the learning rate. According to Equation [3], at time t, the system was in state s_t, and the agent decided to take action a_t. This resulted in moving the system to state s_{t+1} and obtaining a reward of r_{t+1}. Equation [3] is thus used to find new estimates for the Q-values for a new iteration as a function of the values from the previous iteration. The algorithm typically would go through several iterations until it converges to the optimal values of the Q-estimates.

EXPERIMENT SETUP

As was previously mentioned, the objective of the experiment presented in this paper is to have an agent that is capable of recognizing the state of the system and deciding upon a diversion rate at junction J_1. If this diversion rate is followed, the system should eventually move to a state of equilibrium where taking any of the two routes will result in the same travel time. For example, if route A (L1) is totally blocked because of an accident, the agent should guide all motorists to take route B (L2).

State, Action, Reward Definition

State representation is quite important for the agent to learn properly. Ideally, using the CTM, each state is represented by the number of vehicle in each cell. Computationally, it is impossible to use this representation for the states for a RL agent, since this would make the problem state space very large. Finding a good state representation is always a challenging task since it directly affects the efficiency of the learning process of the agent. In this experiment, the state of the system is represented by the difference in the instantaneous time between taking the short route through L1 (route A) and the longer route through L2 (route B). Even with this state representation, there is still a huge number of states in the state space. Therefore, discretizing the state space was also an essential step. Based on some empirical experiments, the state space was discreteized to a finite number of states. Table 1 shows how the state was discretized based upon the difference in instantaneous travel time between the longer route, route B, and the shorter route, route A.

Time difference (dif) in minutes	State code	Time difference (dif) in minutes	State code
0 < dif < 3	0	0 > dif > -3	-10
3 < dif < 7	1	-3 > dif >-7	-1
7 < dif < 14	2	-7 > dif > -14	-2
14 < dif < 24	3	-14 > dif > -24	-3
24 < dif < 40	4	-24 > dif > -40	-4
40 < dif < 90	5	-40 > dif > -90	-5
90 < dif < 125	6	-90 > dif > -125	-6
125 < dif < 250	7	-125 > dif > -250	-7
250 < dif < 375	8	-250 > dif > -375	-8
375 < dif	9	-375 > dif	-9

Table 1: State Space

As can be seen, State 0, for example, refers to the case when both routes are running at free flow speed. For this case, the difference in travel time between routes B and A is in the range of +3 minutes, since route B is 2.0 miles longer than route A. On the other hand, state -9 refers to the case when the route A (the shorter route) is extremely congested (totally closed), while the longer route is doing fine. In our experiments, the instantaneous time is determined from speed sensor readings along the two routes.

For actions, ideally the diversion rate is a real number between 0 and 100%; which means an infinite set of actions. In this experiment, the set of actions were reduced to only six actions. Table 2 indicates the six different actions used in this experiment.

Action code	Action meaning
0	Divert 100% of the flow to L1 Divert 0% of the flow to L2
1	Divert 80% of the flow to L1 Divert 20% of the flow to L2
2	Divert 60% of the flow to L1 Divert 40% of the flow to L2
3	Divert 40% of the flow to L1 Divert 60% of the flow to L2
4	Divert 20% of the flow to L1 Divert 80% of the flow to L2
5	Divert 0% of the flow to L1 Divert 100% of the flow to L2

Table 2: Action Set

For the SARSA algorithm, all the values of Q(s,a) are initialized to zero. The agent keeps deciding on what actions to take in the different states the agent encounters. The environment will respond with a positive reward of 1 only when the system is in either state 0 or –10. Otherwise, the agent gets a reward of -1. In other words, the goal of the agent is to take the proper action to ensure that the instantaneous difference in time between the two routes does not exceed 3 minutes; i.e. reaching (or being close to) the state of equilibrium. In the experiment, the authors simulated running the system for around 90 hours of operation. At the 30^{th} minute, an accident was introduced at link L1. The accident lasted for around 6 hours causing the shorter route to be completely congested. It was expected that the agent would learn by itself from interacting with the system that the best action in being in such a state is to divert all the traffic to the longer route.

RESULTS AND DISCUSSIONS

The results of the experiment show that the agent managed to learn the right actions for the extreme states but failed to learn the proper actions for the in-between states. For the state of complete congestion on link L1 (state -9), the agent learned that the proper action is to divert 100% of the traffic to the longer route. For the free flow state, where there is no congestion at all (state 0), the agent learned that the best action is to divert all the traffic to the shorter path. But the agent did not learn the proper actions for the in-between states (e.g. states –2 and –4).

After investigating, tracing and analyzing the agent, it was discovered that the system passed through states –2 and –4 for a very short period of time. The system was in these states right after the accident happened then the system moved to the extreme state of –9. The system stayed in state –9 for a good 6 hours. These 6 hours were more than enough for the agent to figure out the best action for this state. After the accident was cleared the system again quickly passed through state –2 and –4 and then went into state 0 and remained in it for the rest of the simulation time (around 80 hours). During this time, the agent managed to learn the best action in the case of free flow speed. To overcome this problem, we are currently, preparing a scenario that will ensure staying in different states for reasonable

amounts of time; minimum of 10 hours of simulation time for each state. We expect that the agent will learn the proper action for each state provided that the agent is given enough time to experiment with this particular state.

Figure 4 shows the convergence to the right action for state –9. The time is represented in seconds*10. Notice that the system got into state –9 after almost one hour of the simulation; 30 minutes after the accident. The agent chose action 0 (diverting all the traffic to the shorter route) the first time the agent encountered state –9. This decision is actually the worst decision that can be made in this situation. But as time passes, the agent learned from its previous mistakes till it converged to the best action at around the 120 minutes from the beginning of the simulation. So, it took the agent only an hour of simulation time to converge to the right answer.

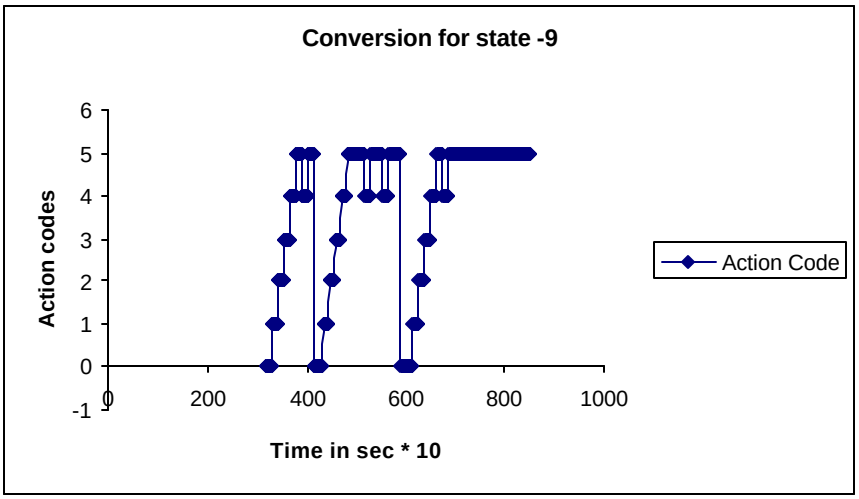

Figure 4: Conversion for State –9

CONCLUSIONS

The results obtained from this very simple experiment are very promising. Designing the right experiment would make the agent learn the proper actions for the different states. In the near future, we are planning to use a neural network to augment the state representation of the system. Using a neural network will allow us to deal with a bigger set of states as well as achieving a smother and continuous representation of the state space. Similarly for actions, a neural net would be much more appropriate than the set of 6 different actions we used in this experiment. In the future we are also planning in experimenting with a more complex network and using more than one agent. We are planning to have a community of cooperating agents. We are also planning to use a microscopic simulation tool (PARAMICS) to model a much more complex network than the one used in this experiment.

ACKNOWLEDGMENTS

This research is being funded by grant number CMS-0133386 from the National Science Foundation (NSF). The authors would like to thank NSF for their support.

REFERENCES

Abdulhai, B. and Kattan L.(2003) "Reinforcement learning: Introduction to theory and potential for transport applications" *Canadian Journal of Civil Engineering*, Volume 30, Number 6 (11), 981-991.

Abdulhai, B., Pringle, P., and Karakoulas, G.J. (2003) "Reinforcement Learning for True Adaptive Traffic Signal Control. *ASCE Journal of Transportation Engineering*, Vol. 129(3), pp. 278-284.

Abdulhai, B. and Pringle, P. (2003). "Autonomous Multiagent Reinforcement Learning –5 GC Urban Traffic Control." A paper presented at the *2003 Annual Transportation Research Board Meeting*, Washington, D.C.

Bertsekas, Dimitri P. (2000*). Dynamic Programming and Optimal Control*. Athena Scientific, Massachusetts USA, 502 pp.

Choy, M. C., Cheu, R.L., Srinivasan, D., and Logi, F. (2003). Real-Time Coordinated Signal Control Using Agents with Online Reinforcement Learning. A paper presented at the *2003 Annual Transportation Research Board Meeting*, Washington, D.C.

Daganzo, C.F. (1994). "The Cell Transmission Model: A Dynamic Representation of Highway Traffic Consistent with the Hydrodynamic." *Transportation Research*, Vol 28B, No. 4, 269-287.

Daganzo, C.F. (1995). "The Cell Transmission Model, Part II: Network Traffic." *Transportation Research*, Vol 29B, No. 2, 79-93.

Kaelbling, Leslie P., Littman, Michael L., and Moore, Andrew W. (1996) "Reinforcement Learning: A Survey." *Journal of Artificial Intelligence Research*, (4) 237-285.

Kretchmar, R. Matthew (2000). *A Synthesis Of Reinforcement Learning And Robust Control Theory*. Ph.D. Diss., computer Science, Colorado Sate University, Fort Collins, Colorado, 139 pp.

Lighthill, M.J. and Whitham, J.B. (1955) "On Kinematic Waves. I: Flow movement in long rivers; II. A theory of traffic flow on long crowded roads." *Proc. Royal Society* A, 229, pp. 281-345

Russell, S. and Norving, P.(2003) *Artificial Intelligence A Modern Approach.* Second Edition, Prentice Hall.

Sutton, Richard S. and Barto, Andrew G. (2000*). Reinforcement Learning*. MIT Press, Massachusetts USA, 322 pp.

Wang, Y, Papageorgiou, M. and A. Messmer. (2003) "A Predictive Feedback Routing Control Strategy for Freeway Network Traffic." *Transportation Research Record 1312*, TRB, National Research Council, Washington, D.C, pp. 21-43.