Research Journal of Applied Sciences, Engineering and Technology 6(16): 2943-2950, 2013
DOI:10.19026/rjaset.6.3676
ISSN: 2040-7459; e-ISSN: 2040-7467
© 2013 Maxwell Scientific Publication Corp.
Submitted: January 02, 2013 Accepted: January 31, 2013 Published: September 10, 2013

Research Article

Urban Traffic Control Using Adjusted Reinforcement Learning in a Multi-agent System

Mahshid Helali Moghadam and Nasser Mozayani
Computer Engineering Department, Iran University of Science and Technology, Tehran, Iran

Abstract: Dynamism, continuous changes of states and the necessity to respond quickly are the specific characteristics of the environment in a traffic control system. Proposing an appropriate and flexible strategy to meet the existing requirements is always an important issue in traffic control. This study presents an adaptive approach to control urban traffic using multi-agent systems and a reinforcement learning augmented by an adjusting pre-learning stage. In this approach, the agent primarily uses some statistical traffic data and then uses traffic engineering theories for computing appropriate values of the traffic parameters. Having these primary values, the agents start the reinforcement learning based on the basic calculated information. The proposed approach, at first finds the approximate optimal zone for traffic parameters based on traffic engineering theories. Then using an appropriate reinforcement learning, it tries to exploit the best point according to different conditions. This approach was implemented on a network in traffic simulator software. The network was composed of six four phased intersections and 17 two lane streets. In the simulation, pedestrians were not considered in the system. The load of the network is defined in terms of Origin-Destination matrices whose entries represent the number of trips from an origin to a destination as a function of time. The simulation ran for five hours and an average traffic volume was used. According to the simulation results, the proposed approach behaved adaptively in different conditions and had better performance than the theory-based fixed-time control.

Keywords: Adjusting pre-learning stage, multi-agent system, reinforcement learning, urban traffic control

INTRODUCTION

Traffic control is one of the challenging issues in our world. Today, according to increase in number of vehicles and traffic congestion in the streets and roadways, development of traffic infrastructures could not be a suitable solution to resolve traffic problem in street networks. So, it is necessary to have a system that controls traffic lights optimally and appropriately in different conditions. Classic and common techniques of traffic control like fixed-time control and time-of-day control may act well when there is a certain amount of traffic in the street network (Orcutt, 1993; U.S. Department of Transportation and Federal Highway Administration, 1997). But when model of traffic volume is variable in different hours, such techniques could not be effective appropriately. Thus, it is necessary to propose an approach which makes decisions dynamically in changing conditions. In recent years, different artificial intelligence approaches such as multi-agent systems have widely been favored for traffic control in the field of intelligent transportation systems. Concept of intelligent agents could be adapted to different parts of the system such as traffic lights, cars and pedestrians. Intelligent agents can learn and adapt themselves to different conditions. These agents can also cooperate with each other and control traffic flow more optimally in the network. In many multi-agent systems, reinforcement learning is used to train agents. This learning is very similar to human learning process.

In the field of using intelligent agents and multi-agent systems for traffic control, a great deal of effort has been devoted in recent years. Wiering (2000) used model-based reinforcement learning for traffic light controllers to minimize the overall waiting time of cars. He used car-based value functions to approximate cars waiting time. Bakker *et al*. (2005) improved Wiering's technique by adding some coordination between agents. In both methods, the agent computes its optimal action with respect to local mode. Moriarty *et al*. (1998) reformulated the traffic control into a distributed artificial intelligence task, in which cars coordinated lane changes to maintain desired speeds and reduce total lane maneuvers. In another work, Bingham (2001) used a neural network in fine-tuning the membership functions of a fuzzy traffic signal controller. The neural learning algorithm used was reinforcement learning. In this approach, the rule base was created using expert knowledge. Following the way, Dresner and Stone

Corresponding Author: Mahshid Helali Moghadam, Computer Engineering Department, Iran University of Science and Technology, Tehran, Iran
This work is licensed under a Creative Commons Attribution 4.0 International License (URL: http://creativecommons.org/licenses/by/4.0/).

(2004) proposed a reservation based system for alleviating traffic congestion. In this system, the intersections are outfitted with a wireless communication system and that they use a specific protocol for communicating with oncoming traffic and giving permission for cars to pass. So, Cars must only traverse intersections when allowed to by the protocol, but otherwise are free to decide for them how to drive. Dresner and Stone (2005) improved their proposed reservation-based system by adding more complexities such as possibility of cars U-turns, acceleration in intersections etc. Also, they identified several opportunities created for multi-agent learning on the parts of both classes of agents (intersection managers and driver agents) by their reservation-based mechanism and its protocol (Dresner and Stone, 2006).

In addition to above investigations, Bazzan (2005) proposed one technique which is based on multi-agent systems. In this method, each intersection is modeled as an independent intelligent agent or a player taking part in a dynamic process in which not only agents own local goals but also a global one has to be taken into account. So, all agents eventually move toward achieving the global goal. In this method, concept of evolutionary game theory is mainly applied. In this technique, role of a traffic control manager is also considered which is tasked to make decision on traffic control policies and manage tactically while agents in the intersections are responsible for operational tasks. Bazzan *et al*. (2010) organized agents in groups of limited size. These groups are then coordinated by another agent, a tutor or supervisor. In this study, multi-agent reinforcement learning for control of traffic signals will be implemented in two situations: agents act individually and agents can be ‘‘tutored’’, meaning that the tutor agent will recommend a joint action. Lu *et al*. (2008) integrated *Q*-learning with multiband model to realize adaptive and coordinated signal setting, in which the former optimized split, the latter optimized offset. Based on this integrated model, adaptive and coordinated signal setting for the three-intersection artery was done. Bazzan (2009) stated some challenging issues in “agentification” of a transportation system and presented problems, methods, approaches and practices in traffic engineering (especially regarding traffic signal control); and tried to highlight open problems and challenges so that future research in multi-agent systems could address them.

In this study, traffic signal controllers located at intersections are assumed as autonomous agents. So, street network is seen as a system which is composed of several intelligent agents. A reinforcement learning augmented by an adjusting pre-learning stage was used for agents. These agents receive some statistical traffic information at pre-learning stage and estimate primary traffic parameters according to traffic engineering theories. After calculating primary parameters, the main phase of learning will be started.

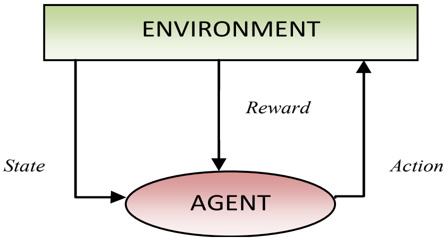


Fig. 1: The interaction of agent with environment in reinforcement learning

REINFORCEMENT LEARNING

Reinforcement learning has attracted rapidly increasing interest in the machine learning and artificial intelligence communities in recent years (Kaelbling *et al*., 1996). The so-called "Reinforcement Learning (RL)" could be introduced as the way of learning through interaction with environment in order to achieve a certain goal. Reinforcement learning is learning way of maximizing a numerical reward signal. The learner is not told which actions to take, as in most forms of machine learning, but instead of mentioned, discover which actions yield the most reward by trying them (Sutton and Barto, 1998). In the standard reinforcement-learning model, an agent is connected to its environment via perception and action, as depicted in Fig. 1 (Alpaydin, 2004). On each step of interaction the agent receives as input, some indication of the current state of the environment. Then, agent chooses an action to generate as output. The action changes the state of the environment and the value of this state transition is communicated to the agent through a scalar reinforcement signal. The agent's behavior should choose actions that tend to increase the long-run sum of values of the reinforcement signal (Kaelbling *et al*., 1996).

Beyond the agent and the environment, there are four main sub-elements of a reinforcement learning system as follows: a policy, a reward function, a value function and, optionally, a model of the environment. A policy is a mapping from perceived states of the environment to actions to be taken when in those states. In some cases the policy may be a simple function or lookup table. A reward function defines the goal in a reinforcement learning problem. It maps each perceived state (or state-action pair) of the environment to a single number, a reward, indicating the intrinsic desirability of that state. A reinforcement learning agent's sole objective is to maximize the total reward it receives in the long run. Whereas a reward function indicates what is good in an immediate sense, a value function specifies what is good in the long run. The value of a state is the total amount of reward an agent can expect to accumulate over the future, starting from that state. The fourth element of some reinforcement learning systems is a model of the environment. The model consists of knowledge of state transition probability function T (s; a; s'), which represents the probability of making a transition from state s to state s' using action a and the reinforcement function R (s; a). Reinforcement learning is primarily concerned with how to obtain the optimal policy when such a model is not known in advance (Kaelbling *et al*., 1996).

The three main categories of RL algorithms are summarized as follows:

- Dynamic programming
- Monte Carlo methods
- Temporal difference methods

The term dynamic programming (DP) refers to a collection of algorithms that can be used to compute optimal policies given a perfect model of the environment as a Markov decision process (MDP). Monte Carlo methods require only experience-sample sequences of states, actions and rewards from on-line or simulated interaction with an environment. Temporal-difference (TD) learning is a combination of Monte Carlo and dynamic programming (DP) ideas. Like Monte Carlo methods, TD methods can learn directly from raw experience without a model of the environment's dynamics. Like DP, TD methods update estimates in part on the basis of other estimates, without waiting for a final outcome.

One of the most important breakthroughs in reinforcement learning was the development of an off-policy TD control algorithm known as *Q*-learning (Watkins, 1989; Sutton and Barto, 1998). *Q*-learning is a form of model-free reinforcement learning. It can also be viewed as a method of asynchronous dynamic programming (DP). It provides agents with the capability of learning to act optimally in Markovian domains by experiencing the consequences of actions, without requiring them to build maps of the domains (Watkins and Dayan, 1992). The *Q*-learning algorithm is shown in procedural form as follows:

```
1.      Initialize $Q$ (s, a)
2.      Observing the current state ($s$)
3.      Repeating the following loop until the goal state
        is reached
3-1.    Selection of an action ($a$) in one of the two
        following modes:
3-1-1.  Randomly (exploration)
3-1-2.  According to $Q$-table which is built until now
        (extraction)
3-2.    Being rewarded by the environment ($r$)
3-3.    Receiving the new state of the environment ($s'$)
3-4.    Changing the value in $Q$-table according to
        following expression:
```

$$Q\,(\mathrm{s, a}) \leftarrow \alpha\,(\mathrm{r} + \gamma \max Q\,(\mathrm{s', a'})) + (1 - \alpha)\,Q\,(\mathrm{s, a}) \quad (1)$$

```
3-5.    Taking the next state as the current state (s ← s')
```

In the *Q*-learning, the learned action-value function *Q*, directly approximates *Q**, the optimal action-value function, independent of the policy being followed. This dramatically simplifies the analysis of the algorithm and enabled early convergence proofs (Mitchell, 1997; Sutton and Barto, 1998; Szepesvári, 2010). If each action selected in specific state an infinite number of times on an infinite run and α is decayed appropriately, the *Q* values will converge with probability 1 to *Q** (Watkins, 1989; Tsitsiklis, 1994; Kaelbling *et al*., 1996). When the *Q* values are nearly converged to their optimal values, it is appropriate for the agent to act greedily, taking, in each situation, the action with the highest *Q* value. During learning, however, there is a difficult exploitation versus exploration trade-off to be made.

In this algorithm, $\alpha \in [0, 1]$ is the learning rate and determines that to what extent the newly obtained information will override the old information. The learning rate controls how fast the estimates will be changed. Value 1 for this rate causes the agent only to consider the latest information while value 0 may cause the agent not to learn anything. $\gamma \in [0, 1]$ is discount factor to determine the importance of future rewards. Value 0 shows that the agent only considers the current reward. On the other hand, approaching to value 1 will make it strive for a long-term high reward (Kaelbling *et al*., 1996; Szepesvári, 2010).

URBAN TRAFFIC CONTROL BASED ON ADJUSTED REINFORCEMENT LEARNING

The proposed approach for traffic signals control is based on use of a reinforcement learning which is accompanied with an adapting pre-learning stage. In this approach, the traffic network is considered as a system composed of intelligent agents. Traffic signal controllers located at intersections are assumed as autonomous agents. In the proposed approach, before start of the learning process, agents gain some traffic information and calculate the approximate optimal zone for traffic parameters including cycle length and green time of phases, based on traffic engineering theories and then begin to learn. At first, agents acquire some statistical traffic information including rates of flow and saturation flows of the network streets. The rate of flow is defined as the number of vehicles passing a point on a highway, or a given lane or direction of a highway, during a specified time interval. Rates of flow are generally stated in units of “vehicles per hour”. Saturation flow is calculated by assuming that every vehicle (in a given lane) consumes an average of “h” seconds of green time to enter the intersection. The relation between the saturation follow and rate of flow is as follows:

$$\mathrm{s} = \frac{3600}{\mathrm{h}} \qquad (2)$$

where s is the saturation flow rate, vehicles per hour of green per lane (veh/hg/ln) and h is the saturation

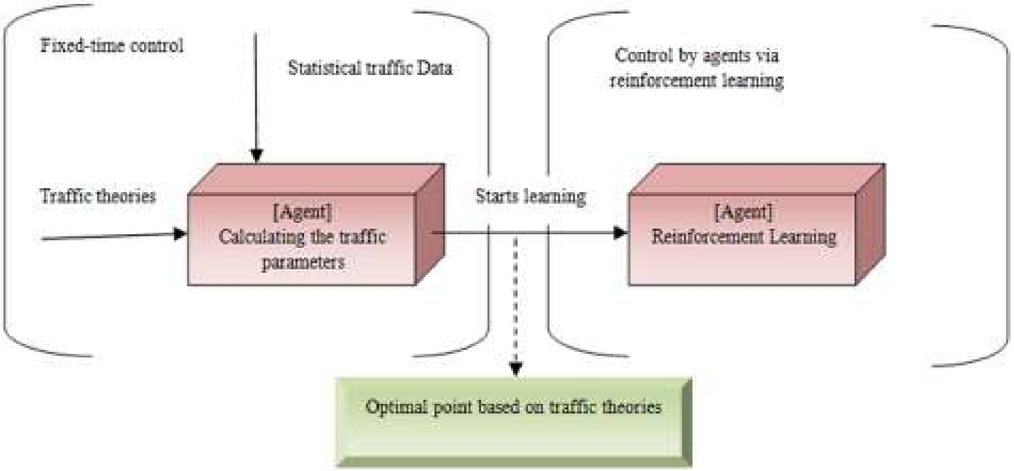

Fixed-time control
Statistical traffic Data
Traffic theories
[Agent]
Calculating the traffic parameters
Starts learning
Control by agents via reinforcement learning
[Agent]
Reinforcement Learning
Optimal point based on traffic theories

Fig. 2: A view of different phases of the proposed approach

headway, seconds/vehicle (s/veh). Headway is defined as the time interval between successive vehicles as they pass a point along the lane, also measured between common reference points on the vehicles. The average headway in a lane is directly related to the rate of flow. In general, this occurs from the fourth or fifth headway position. The constant headway achieved is referred to as the saturation headway, as it is the average headway that can be achieved by a saturated, stable moving queue of vehicles passing through the signal.

Either this statistical information is given to the agents in advance, or is gathered by agents through the methods of collecting statistical data for example monitoring traffic network. Then, according to basic theories of traffic engineering, each agent computes the appropriate cycle length and primary green times for the intersection. Calculating cycle length and related green time intervals is as follows. In calculating cycle length, two types of lost time are considered: start up lost time and clearance lost time. Start up lost time occurs each time a queue starts to move. It is referring to time as drivers react to the green signal and accelerate. Clearance lost time is associated with stopping the queue at the end of the green signal. It is defined as the time interval between the last vehicle's front wheels crossing the stop line and the initiation of the green for the next phase. Clearance lost time is defined as follows:

$$l_2 = Yellow\ time + all\ red\ time - e,\ e = 2 \quad (3)$$

In (3), e is encroachment of vehicles into yellow and all-red time. A default value of 2.0 s is used for e. The total lost time per phase is the sum of start-up lost time and clearance lost time. It is indicated in (4):

$$t_L = l_1 + l_2 \quad (4)$$

So the total lost time per cycle is defined as follows:

$$L = N \times t_L \quad (5)$$

where, N is the number of phases in the cycle. Then the cycle length will be calculated as follows:

$$C = \frac{L}{(1-\frac{V_C}{[(\frac{3600}{h})\times PHF\times\frac{v}{c}]})} \quad (6)$$

where, Vc is sum of critical lane volumes (vehicle/hour), h is saturation headway (second/vehicle), PHF is the peak hour factor to estimate the flow rate in the worst hours and v/c is the desired volume to capacity ratio. Accordingly, the total effective green time and effective green time for each phase will be calculated as follows (Roess *et al*., 2004; Currin, 2012):

$$G = C - L \quad (7)$$

$$g_i = G \times (\frac{y_i}{Y}) \quad (8)$$

where,
G : The Total effective green time
g_i : Effective green time for phase i

So the actual green time for each phase is defined as follows:

$$G_i = g_i - (Yellow + all\ red\ time) + t_{Li} \quad (9)$$

Afterwards, the calculated cycle length and green time values are set in the intersection. The agent starts reinforcement learning phase after the first cycle length. It gradually learns to control the traffic in the network in different conditions adaptively. Figure 2 shows a view of different phases of the proposed approach.

In the reinforcement learning phase, the agent computes the reward at the end of every cycle length. Then it updates *Q*-table and selects the next action. In the learning, 24 states have been defined. These states indicate the relations between traffic counts for the streets in the intersection. (Traffic counts, also called traffic volumes). For instance, State I shows the

Table 1: List of possible actions

Actions	Street A Green Time (SAGT)	Street B Green Time (SBGT)	Street C Green Time (SCGT)	Street D Green Time (SDGT)
Action I	SAGT+0	SBGT+0	SCGT+0	SDGT+0
Action II	SAGT+0	SBGT-1	SCGT+1	SDGT+0
Action III	SAGT+0	SBGT+1	SCGT-1	SDGT+0
Action IV	SAGT-1	SBGT+0	SCGT+0	SDGT+1
Action V	SAGT-1	SBGT-1	SCGT+1	SDGT+1
Action VI	SAGT-1	SBGT+1	SCGT-1	SDGT+1
Action VII	SAGT+1	SBGT+0	SCGT+0	SDGT-1
Action VIII	SAGT+1	SBGT-1	SCGT+1	SDGT-1
Action IX	SAGT+1	SBGT+1	SCGT-1	SDGT-1
Action X	SAGT-1	SBGT+1	SCGT+0	SDGT+0
Action XI	SAGT+1	SBGT-1	SCGT+0	SDGT+0
Action XII	SAGT+0	SBGT+0	SCGT-1	SDGT+1
Action XIII	SAGT+1	SBGT-1	SCGT-1	SDGT+1
Action XIV	SAGT+0	SBGT+0	SCGT+1	SDGT-1
Action XV	SAGT-1	SBGT+1	SCGT+1	SDGT-1
Action XVI	SAGT-1	SBGT+0	SCGT+1	SDGT+0
Action XVII	SAGT+1	SBGT+0	SCGT-1	SDGT+0
Action XVIII	SAGT+0	SBGT+1	SCGT+0	SDGT-1
Action XIX	SAGT+0	SBGT-1	SCGT+0	SDGT+1

following status. Traffic count on street A is greater than the street B and Also Traffic count on street B is greater than street C. In other hand, traffic count on street C is greater than street D. Accordingly, State II is representative of a state which traffic volume in street A is greater than the street B and the traffic volume in the street B is greater than the street D and traffic volume in the street D is also greater than the street C. So the number of states equals to the number of permutations of the list of four streets in the intersection.

In this approach, 19 possible actions are defined for each agent. Accordingly, the agent can change the existing green intervals by adding or subtracting 1s in such a way that the cycle length remains constant. Also the agent can use the existing intervals without any changes. For example, Action I does not change the existing intervals and keeps them constant for the next cycle length. Action II adds one second to Street C Green Time but subtracts one second from Street B Green Time. Table 1 shows the list of possible actions.

In the learning phase, a minimum value is also considered for each green time in order to prevent the agent from reducing it any more. It means if one of the green times has minimum value and the agent has selected an action that reduces it more, the agent will not be allowed to apply this action and should select action I (The action that doesn't change the green intervals). In this approach, the received reward by each agent is calculated as follows:

$$r = -\frac{\text{total existing cars in the streets of the intersection}}{\text{capacity of the intersection streets}} \quad (10)$$

Where the capacity of each street is the maximum number of cars which can be placed through the given street.

CASE STUDY

In this study, the traffic simulator software, Aimsun V6.1 and its AAPI environment was used for implementing the proposed approach on a sample network. Aimsun is an integrated transport modeling software, developed and marketed by TSS. It is used to improve road infrastructure and design urban environments for vehicles and pedestrians. Aimsun integrates three types of transport models (macroscopic, mesoscopic and microscopic) into one software application. It can be customized using Python, Qt and C++. It has an AAPI editor for programming in C++ (Aimsun 6.1 Users Manual, 2009). In this study, the proposed approach was implemented on a network composed of six four phased intersections and 17 two lane streets. Figure 3 shows the sample network. The number of installed traffic lights in each intersection equals to the number of its phases. After designing the network structure, the statistical features of the network such as traffic volumes on the streets, the way of car injection into the network and the way of distributing the incoming traffic in the network must be specified.

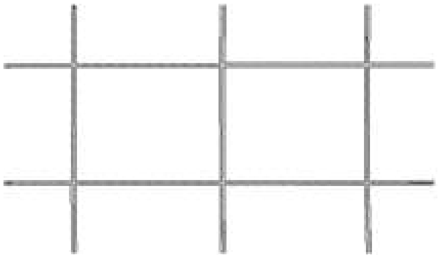
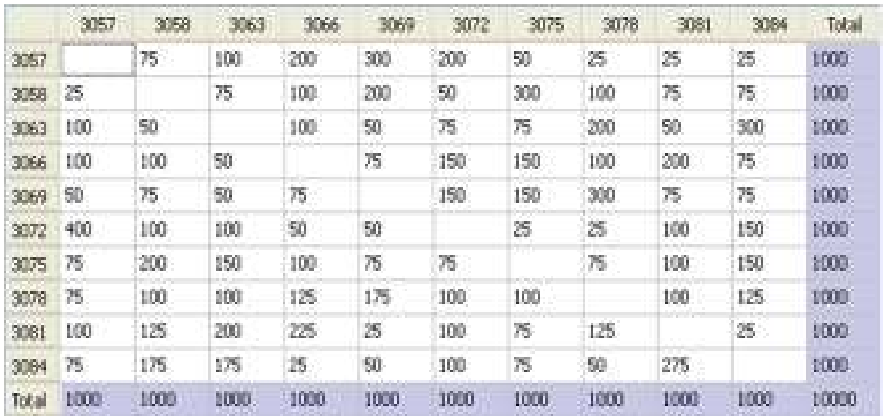


Fig. 3: The sample network

There are two ways for specifying traffic demand in the network:

- Origin and destination matrix
- Using traffic state concept (Aimsun 6.1 Users Manual, 2009)

In this study, the load of the network is defined in terms of Origin-Destination matrices whose entries



	3057	3058	3063	3066	3069	3072	3075	3078	3081	3084	Total
3057		75	100	200	300	200	50	25	25	25	1000
3058	25		75	100	200	50	300	100	75	75	1000
3063	100	50		100	50	75	75	200	50	300	1000
3066	100	100	50		75	150	150	100	200	75	1000
3069	50	75	50	75		150	150	300	75	75	1000
3072	400	100	100	50	50		25	25	100	150	1000
3075	75	200	150	100	75	75		75	100	150	1000
3078	75	100	100	125	175	100	100		100	125	1000
3081	100	125	200	225	25	100	75	125		25	1000
3084	75	175	175	25	50	100	75	50	275		1000
Total	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	10000

Fig. 4: A sample of origin and destination matrix

Table 2: Traffic data for intersections I and II

	Intersection I				Intersection II			
	St. A	St. B	St. C	St. D	St. A	St. B	St. C	St. D
Flow	475	257	308	321	454	379	384	162
Saturation flow	1800	1800	1800	1800	1800	1800	1800	1800
Ratio: yi	0.26	0.14	0.17	0.18	0.25	0.21	0.21	0.09

Table 3: Traffic data for intersections III and IV

	Intersection III				Intersection IV			
	St. A	St. B	St. C	St. D	St. A	St. B	St. C	St. D
Flow	345	484	296	227	360	275	368	239
Saturation flow	1800	1800	1800	1800	1800	1800	1800	1800
Ratio: yi	0.19	0.27	0.16	0.13	0.20	0.15	0.20	0.13

Table 4: Traffic data for intersections V and VI

	Intersection V				Intersection VI			
	St. A	St. B	St. C	St. D	St. A	St. B	St. C	St. D
Flow	346	294	279	322	344	333	266	227
Saturation flow	1800	1800	1800	1800	1800	1800	1800	1800
Ratio: yi	0.19	0.16	0.16	0.18	0.19	0.19	0.15	0.13

Table 5: Cycle length and green time of phases in the intersections

	Intersection I	Intersection II	Intersection III	Intersection IV	Intersection V	Intersection VI
Cycle length (s)	100	120	100	80	80	60
Actual green time of phase A (s)	31	36	23	20	19	14
Actual green time of phase B (s)	17	30	32	15	16	14
Actual green time of phase C (s)	20	30	19	20	16	11
Actual green time of phase D (s)	20	12	14	13	17	9

represent the number of trips from an origin to a destination as a function of time. This network has 10 origin-destination centroids. Vehicles are generated at each origin centroid and input into the network. Then, vehicles are distributed along the network following shortest paths between origin and destination centroids. Finally, vehicles exit the network via the destination or sink centroid (Aimsun 6.1 Users Manual, 2009). Figure 4 shows a sample of origin and destination matrix.

In the simulation, pedestrians were not considered in the system. The simulation ran for 5 h and an average traffic volume was used. Time intervals between two consecutive vehicle arrivals (headway) at input sections are sampled from an exponential distribution. In the Exponential distribution, the default generation model in Aimsun, the mean input flow (in vehicles/second) is λ and the mean time headway is calculated as 1/λ seconds (Aimsun 6.1 Users Manual, 2009).

For implementing the proposed approach on the network, first of all, the necessary statistical data of the traffic network including flows and saturation flows will be collected. For doing this, an arbitrary fixed-time control was initially applied to the network for one hour and agents collected the data of flows and saturation flows of the streets. In the simulation, it assumes the saturation flow is equal to the capacity of the street. Table 2 to 4 shows the flows and saturation flows of the streets.

The calculated cycle length and actual green time of phases for the intersections are shown in Table 5. For

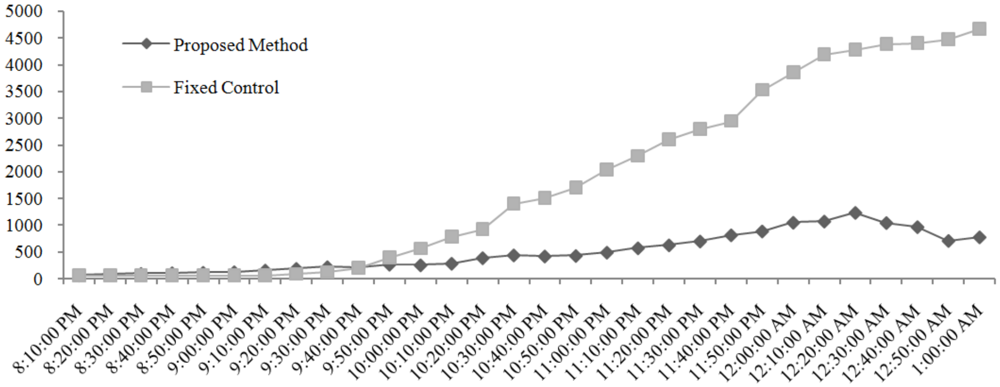


Fig. 5: Comparison of stop time which is resulting from the fixed-time control with the stop time from implementing the proposed approach

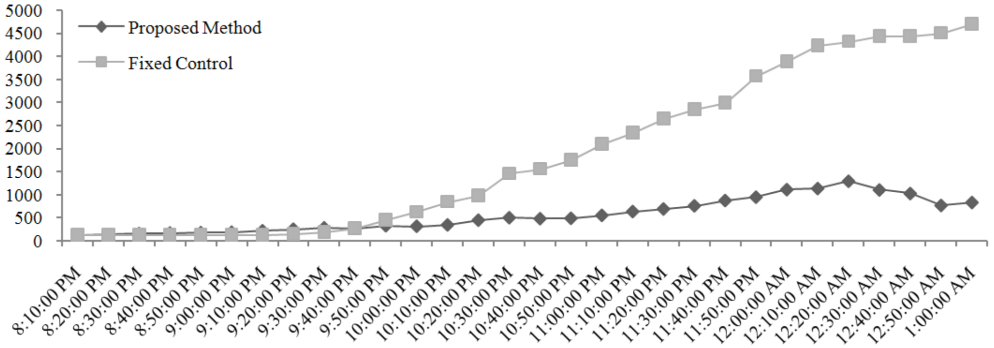


Fig. 6: Comparison of travel times obtained with two methods

calculating these parameters, it assumes that start up lost time l_1 = 2s and clearance lost time l_2 = 1s. Also “yellow time” and “all red time” have been considered as zero and 3s, respectively.

The results of the proposed approach have been compared with the results of using a basic control method for traffic lights, theory-based fixed-time control. In this fixed-time control, all the parameters including cycle length and actual green time of phases are calculated based on theoretical principles. They will be constant during the control. The resultant outcomes from the simulation show that the proposed approach has better performance than the fixed-time control. In other words, the proposed approach is able to behave adaptively in different conditions. Figure 5 shows the comparison of stop time which is resulting from fixed-time control with the obtained stop time from implementing the proposed approach on the same network. Figure 6 also shows travel time in both methods. These figures signify that at the beginning of simulation, both methods have identical behavior but gradually the proposed learning indicates clearly its superiority in performance.

CONCLUSION

The multi agent systems and reinforcement learning have an appropriate potential for being applied to traffic control. The proposed approach, in the first stage, finds the approximate optimal zone for traffic parameters including cycle length and green time of phases, based on traffic engineering theories. Then, using appropriate reinforcement learning, it tries to exploit the best point according to different conditions. In other words, with little dynamic changes in the traffic condition, it fluctuates on the base zone to find the best point. By comparing the results, the proposed approach, behaves adaptively in the dynamic changing environment. So it prevents the network saturation and occurring critical situations and could deal with traffic changing appropriately. Using the basic data resultant from traffic engineering theories as information infrastructure, causes the learner agent explore in a semi-optimal zone to find the best point. Since, the environment in traffic control system is dynamic and sensitive to selected action, selecting unsuitable signal timing even for a short period of time, can cause saturation and critical situation in the network. So using the results of traffic theories as a basis for reinforcement learning phase helps to improve the performance of reinforcement learning. The results of simulation show that the proposed approach controls traffic volume in street network more dynamically and flexibly than fixed-time control. In this approach, control of traffic lights is adaptive to changing conditions. Benefitting the basic information from traffic theories, as an information basis in reinforcement learning, is considered as a special advantage of the proposed approach.

For future research, the following directions could be suggested:

- Establishing coordination between agents and considering the traffic condition of the adjacent intersections in the control approach
- Suggesting a suitable approach for creating green waves in some intersections (A green wave is an intentionally induced phenomenon in which a series of traffic lights (usually three or more) are coordinated to allow continuous traffic flow over several intersections in one main direction)

REFERENCES

Aimsun 6.1 Users Manual, 2009. Draft Version-October.

Alpaydin, E., 2004. Introduction to Machine Learning. MIT Press, Cambridge, Mass, pp: 373-377.

Bakker, B., M. Steingrover, R. Schouten, E. Nijhuis and L. Kester, 2005. Cooperative multi-agent reinforcement learning of traffic lights. Proceeding of Workshop on Cooperative Multi-Agent Learning, European Conference on Machine Learning (ECML'05), Portugal.

Bazzan, A.L.C., 2005. A distributed approach for coordination of traffic signal agents. Auton. Agent. Multi-Ag., 10(1): 131-164.

Bazzan, A.L.C., 2009. Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. Auton. Agent. Multi-Ag., 18(3): 342-375.

Bazzan, A.L.C., D. De Oliveira and B.C. Da Silva, 2010. Learning in groups of traffic signals. Eng. Appl. Artif. Intell., 23(4): 560-568.

Bingham, E., 2001. Reinforcement learning in neurofuzzy traffic signal control. Eur. J. Oper. Res., 131(2): 232-241.

Currin, T.R., 2012. Introduction to Traffic Engineering: A Manual for Data Collection and Analysis. 2nd Edn., Cengage Learning, Andover, pp: 45-80.

Dresner, K. and P. Stone, 2004. Multiagent traffic management: A reservation-based intersection control mechanism. Proceeding of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems. New York, USA, pp: 530-537.

Dresner, K. and P. Stone, 2005, Multiagent traffic management: An improved intersection control mechanism. Proceeding of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems. Utrecht, Netherlands, pp: 471-477.

Dresner, K. and P. Stone, 2006. Multiagent Traffic Management: Opportunities for Multiagent Learning. In: Tuyls, K. *et al*. (Eds.), Lecture Notes in Artificial Intelligence. Springer Verlag, Berlin, 3898: 129-138.

Kaelbling, L.P., M.L. Littman and A.W. Moore, 1996. Reinforcement learning: A survey. J. Artif. Intell. Res., 4: 237-285.

Lu, S., X. Liu and S. Dai, 2008. Adaptive and coordinated traffic signal control based on Q-learning and Multiband model. Proceeding of IEEE Conference on Cybernetics and Intelligent Systems, pp: 765-770.

Mitchell, T., 1997. Machine Learning. 1st Edn.,, McGraw-Hill Education (ISE Editions), Boston, MA, pp: 379-399.

Moriarty, D.E., S. Handley and P. Langley, 1998. Learning distributed strategies for traffic control. Proceeding of the 5th International Conference of the Society for Adaptive Behavior. Zurich, Switzerland, pp: 437-446.

Orcutt, F.L., 1993. The Traffic Signal Book. Prentice Hall, Englewood Cliffs, New Jersey, pp: 59-65.

Roess, R.P., E.S. Prassas and W.R. Mcshane, 2004. Traffic Engineering. 3rd Edn., Pearson Education, Upper Saddle River, NJ, pp: 455-504.

Sutton, R.S. and A.G. Barto, 1998. Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, pp: 133-157.

Szepesvári, C., 2010. Algorithms for Reinforcement Learning. Morgan & Claypool, San Rafael, pp: 45-65.

Tsitsiklis, J.N., 1994. Asynchronous stochastic approximation and Q-learning. Mach. Learn., 16(3): 185-202.

U.S. Department of Transportation and Federal Highway Administration, 1997. Advanced Transportation Management Technologies, pp: 3: 1-3: 28.

Watkins, C.J.C.H., 1989. Learning from delayed rewards. Ph.D. Thesis, King's College, Cambridge, UK.

Watkins, C.J.C.H. and P. Dayan, 1992. Q-learning. Mach. Learn., 8(3): 279-292.

Wiering, M., 2000. Multi-agent reinforcement learning for traffic light control. Proceeding of the 17th International Conference on Machine Learning and Applications. USA, pp: 1151-1158.