2014 IEEE 17th International Conference on
Intelligent Transportation Systems (ITSC)
October 8-11, 2014. Qingdao, China

Multi-agent Reinforcement Learning for Traffic Signal Control

Prabuchandran K.J.[1], Hemanth Kumar A.N[1], Shalabh Bhatnagar[1], *Senior Member, IEEE*

***Abstract*— Optimal control of traffic lights at junctions or traffic signal control (TSC) is essential for reducing the average delay experienced by the road users amidst the rapid increase in the usage of vehicles. In this paper, we formulate the TSC problem as a discounted cost Markov decision process (MDP) and apply multi-agent reinforcement learning (MARL) algorithms to obtain dynamic TSC policies. We model each traffic signal junction as an independent agent. An agent decides the signal duration of its phases in a round-robin (RR) manner using multi-agent Q-learning with either ϵ-greedy or UCB [3] based exploration strategies. It updates its Q-factors based on the cost feedback signal received from its neighbouring agents. This feedback signal can be easily constructed and is shown to be effective in minimizing the average delay of the vehicles in the network. We show through simulations over VISSIM that our algorithms perform significantly better than both the standard fixed signal timing (FST) algorithm and the saturation balancing (SAT) algorithm [15] over two real road networks.**

***Index Terms*— traffic signal control, multi-agent reinforcement learning, Q-learning, UCB, VISSIM.**

I. Introduction

Traffic congestion arising due to constantly increasing urban traffic volumes is a serious concern that needs to be addressed by the transportation research community. Initial attempts to improve the road infrastructure could not alleviate this problem due to limitations on financial and spatial resources. This led to the alternative approach of intelligently controlling the traffic lights at junctions. In this approach the uncertainty in the traffic flow is first modeled appropriately. Then, either the green signal duration of the phases is optimized, or the order of the phase sequence at the intersections is controlled in order to maximize the traffic flow. There are numerous research studies using this approach (see [12]) and in our work, we follow the same to improve the traffic flow.

Real time traffic evolves according to a complex stochastic process. Modeling the problem of optimizing the green signal duration of phases in a junction along with the real time traffic is crucial for maximizing the traffic flow. Markov decision process (MDP) offers a good framework to achieve this goal. However, the dimensionality of the state and action space grows exponentially with the number of intersections (junctions) in the road network. And thus, one may have to resort to approximation methods to solve the MDP. Multi-agent reinforcement learning (MARL) provides us with a convenient mathematical platform to attack this problem. Reinforcement learning (RL) methods are well-suited here because they are online in nature and learn good control strategies from experience. Multi-agent systems are also well-suited because they help in controlling the dimensionality of the state and action spaces, as each agent views only a portion of the state space and controls the same.

There is an enormous body of research aimed at minimizing the delay experienced by the road users through MARL methods. A survey of the feasibility and applicability of the multi-agent system for the TSC problem is presented in [5]. It describes the various approaches deployed to optimize the traffic and demonstrates that the TSC problem provides an excellent testbed for the MARL approach. A max-plus algorithm as a coordination strategy in a decentralized setting is proposed in [11]. The reward structure incorporates the max-plus algorithm, thereby making agents gain incentive for coordination. Multi-agent Q-learning for large traffic networks has been proposed in [1] where Q-factors are updated based on local information at each intersection.

RL with function approximation (FA) is studied in [13] for dealing with large state-action spaces. This alleviates the curse of dimensionality effect but poses the problem of feature selection. Also, convergence guarantees for RL with FA can be provided only under stronger assumptions. In [4], vehicle-based representation has been used to construct the model of the environment from the samples and dynamic programming is applied to estimate the optimal value function. A multi-agent system with RL is described in [2] where two kinds of agents are employed - a central agent and an out-bound agent. The central agent learns to control from Q-learning with value function approximation while the out-bound agent follows the longest queue first algorithm. As FA does not guarantee convergence for multi-agent systems and the policies obtained are sub-optimal, the authors of [2] mention a few techniques to enhance the performance of FA. Q-learning based acyclic signal control system for a single intersection with multiple phases is studied in [9] using three different state representations; however, the actions need not generate a round robin (RR) phase sequence and reward calculations involved in the Q-value update are not simple.

In [8], a TSC algorithm based on adaptive dynamic programming (ADP) is presented and its performance compared with the conventional pre-timed, actuated control and an RL algorithm. ADP based on temporal difference learning and perturbation learning using FA for the case of an isolated junction is analyzed in [6]. The TSC problem using coordination graphs to describe the dependencies between agents is modelled in [10] and the max-plus algorithm is

[1] Department of Computer Science and Automation, Indian Institute of Science, Bangalore-560012, India. {prabu.kj,hemanth.kumar, shalabh} @csa.iisc.ernet.in

This work was partially supported by projects from Xerox Corporation, USA, Robert Bosch Centre, IISc, and Department of Science and Technology.

applied to estimate the optimal joint action.

A collaborative RL algorithm is described in [16] where the Q-learning agents base their action selection according to the Boltzmann action selection technique. In [16], despite having an adaptive RR scheme for the phase sequence, the same is not strictly RR because the action space there consists of zero second duration phases and this allows some phases to be skipped in the phase sequence. Also, the traffic patterns considered are steady and uniform in nature. In [19], three models to determine the optimal signal timings have been proposed and their methodology has been tested on an isolated intersection setting. A self-organizing control of traffic lights using a counter is studied in [7]. In this approach, the traffic light in a phase is set to green when the number of vehicles waiting for the phase to turn green crosses a certain threshold.

In this paper, we formulate the problem of intelligently controlling the traffic lights in a road network, namely the TSC problem, as a multi-agent coordination problem. We model here each traffic intersection as an agent. Each agent updates its Q-factors (see section III) using Q-learning with either an ϵ-greedy or a UCB based exploration strategy using a feedback cost signal obtained from its neighbours. Then, based on the learnt Q-factors, the agent decides the green duration of the phases at the multi-phase intersection that it controls. The switching order of the phases is RR. This is guided more by the psychology of the motorists who normally prefer RR scheduling of traffic lights. We show that multi-agent learning yields a policy that has significantly better performance than both the FST and the SAT algorithms.

- It is able to tackle the curse of dimensionality as each agent only handles the portion of the state space associated with its own intersection.
- The cost signal used by the agents to update Q-factors is simple but effective in arriving at a near-optimal policy.
- The modelling of action space results in an RR scheduling of the phases at each intersection. Such a scheduling is more practical because it helps road users know about their turn in advance except for the duration of the green time which varies based on the traffic condition.
- The first phase of the RR schedule for each intersection is initially set randomly to one of the possible phases in our algorithms. As the learning progresses, the agents at intersections interact with one another asynchronously to achieve a self organizing behaviour.

The rest of the paper is organized as follows. In section II, we formulate the TSC problem as an MDP. Section III then presents the MARL algorithm under both ϵ-greedy and UCB based exploration for solving the MDP. Section IV provides a performance comparison of our algorithm with the FST and the SAT algorithms. Finally, Section V provides the concluding remarks.

II. Traffic Signal Control problem as an MDP

A traffic network consists of many junctions. Let J denote the number of junctions in the network. At each junction, there are a certain number of incoming and outgoing traffic lanes. In particular, by an incoming lane, we mean the lane along which the vehicles approach the junction. It can be noted that the sets of all the incoming lanes at individual junctions in the network partitions all the lanes present in the road network. A phase is the duration of time for which a group of signal heads is set to green and the phase configuration is the traffic light configuration during the phase. In a fixed signal timing (FST) algorithm, the various phases are visited in an RR fashion and the duration of each phase is pre-calculated based on the estimated traffic. Due to the dynamic and uncertain nature of the traffic arrivals, the FST control policy leads to large waiting time for the road users and thus, does not minimize the average delay experienced by the road users. Even though it is widely used, this policy often leads to traffic congestion. Hence, to minimize the average delay and to avoid traffic congestion, one should resort to policies that dynamically allocate the green time to each phase. The SAT algorithm, which is an adaptive TSC technique inspired from the SCATS algorithm, dynamically changes the signal timings at each decision point in small steps. It computes the volume of traffic in all the lanes at a junction from the detector loops in each lane placed near the stop-lines of the junction. It adjusts the signal timings of the phases in small steps based on the traffic volume calculated (see [16]). Even though SAT is a simple adaptive scheme it is not very effective (in the sense that a hand-tuned controller performs better than SAT) in certain traffic conditions due to its slow nature of tuning the phase-timings. Our MARL algorithms, on the other hand, are dynamic in nature while adapting fast to the nature of unknown traffic (see Section IV). The rest of the section is devoted to formulating the TSC problem as an MDP.

An MDP is a controlled Markov process for which the controls or actions are chosen in each state to minimize a certain long-term cost. The MDP framework requires a description of states, actions, and costs. We consider discrete state and action spaces.

A state s^j for a given junction j is a vector of dimension $L+1$, where L denotes the number of incoming lanes in that junction. The ith component of the state vector, $q_i^j,\ i \in \{1, 2 \ldots, L\}$ denotes the queue-length of the traffic in the ith lane at that junction. The last component q_{L+1}^j gives the index of the phase that has to be set green in the RR schedule. The state space of our MDP road network is the cartesian product of the state spaces of the junctions present in the network. Let S^j denote the set of all states at junction j. Then, $S \triangleq \bigtimes_{j=1}^{J} S^j$ (the cartesian product of the S^js)will denote the state space of the MDP. This presents a problem as the state space becomes large rapidly with the number of junctions in the network. In our MARL algorithm, we overcome this problem by considering each traffic signal controller (SC) separately for each junction as an agent. This is equivalent to saying that each agent observes only a portion of the state space and the resulting process is then

a partially observable MDP (POMDP). The computations for any junction are then performed by the SC for that junction.

Despite considerably reducing the size of the state space using a decentralized setting, the cardinality of state space (number of possible states) at any given junction is upper bounded by $(q_{\max})^L$, where $q_{\max}$ denotes the maximum possible queue-length of the vehicles in any lane at that junction and L is the number of lanes there. Note that we do not require all lanes to be of the same length and $q_{\max}$ is only an upperbound on the queue lengths across all lanes. This quantity, still poses the problem of large state space for that junction. Hence, we cluster queue size into three portions - $\{low = 0, medium = 1 \ and \ high = 2\}$. The advantage of this kind of representation is two-fold - we do not require precise information about the queue length as, in real scenarios, it is often difficult to acquire this information, and further this representation controls the size of the state space. In order to segregate the queue size, two sensors may be placed - one at distance D_1 and another at D_2 ($D_1 < D_2$) from the traffic lights on each lane. The traffic is then characterized as $\{low, medium \ and \ high\}$ as follows:

$$q_i^j(t) = \begin{cases} 0, & \text{if } q_i^{j'}(t) < D_1, \\ 1, & \text{if } D_1 \leq q_i^{j'}(t) \leq D_2, \quad i \in \{1, 2, \ldots, L\} \\ 2, & \text{if } q_i^{j'}(t) > D_2. \end{cases}$$

where $q_i^{j'}(t)$ corresponds to the actual congestion level at time t on the ith lane at junction j. The idea is to first trigger the sensor at D_1. If it does not detect vehicles, we infer the queue-length as low. If, on the other hand, sensor at D_1 detects vehicles then we also trigger the sensor at D_2. If the latter sensor does not detect congestion, we infer the queue length as medium. Finally, if the latter sensor also detects congestion, the queue size can be inferred as high. Thus, the state vector s_t^j for a given junction $j \in \{1, 2, \ldots, J\}$ at time t is the vector of segregated queue lengths on each of the incoming lanes at that junction and is denoted by $s_t^j = (q_1^j(t), \ldots, q_{L_j}^j, q_{L_{j+1}}^j(t))^T$, where $q_i^j(t), \ i \in \{1, \ldots, L_j\}$ corresponds to the segregated queue length on lane i at time t, $q_{L_{j+1}}^j(t)$ corresponds to the index of the phase that has to be set green and L_j corresponds to the number of incoming lanes at junction j.

In a similar spirit to the state space reduction by considering each junction separately, we consider the action spaces individually for each of the junctions to avoid action space explosion. The action of an agent at a given junction is to decide the duration of phases visited in the RR fashion. The phases visited follow a prescribed RR order that is known in advance. At each junction as discussed before, the last component of the state vector keeps track of the phase whose green time has to be set at the current time t and is referred to as the current phase. The action of the agent then corresponds to specifying the duration of the current phase. We discretize the action space A into three values - $\{low, medium \ and \ high\}$ to handle the curse of dimensionality in the action space. Note the abuse of notation as the same terms are also used to infer congestion levels. In our experiments we set the three discretized actions (signal timings for the phases) to be $\{low = 10, medium = 20 \ and \ high = 30\}$ seconds. A policy $\pi^j = \{\mu_t^j, t \geq 0\}$ followed at the junction j is a sequence of maps μ_t^j from the state space to the action space such that when the state is s_t^j at instant t, $\mu_t^j(s_t^j)$ specifies the time duration for the current phase. We only consider stationary deterministic policies Π that do not change with time. We thus denote a policy π^j by $\pi^j = \{\mu^j, \mu^j, \ldots\}$, where $\mu^j(s_t^j)$ is the action in state s_t^j. All actions here are assumed feasible in every state. The goal in our setting is to obtain policies for each of the junctions so that the long-term average delay of the road users is minimized.

The cost feedback signal obtained by each agent differs based on the number of neighbouring junctions surrounding it. The immediate cost $c_j(t)$ incurred by an agent at junction $j \in J$ for taking an action a_t^j when the current state is s_t^j and the next state is s_{t+1}^j, is given by

$$c_j(t) = \frac{1}{|N_j|} \sum_{k \in N_j} \sum_{i=1}^{L_k} q_i^k(t+1), \tag{1}$$

where N_j corresponds to the set of neighbouring junctions to junction j and $|N_j|$ its cardinality. This cost would enable each junction to consider the effect of its actions on junctions in its vicinity. The cost $c_j(t)$ measures how the queue-lengths of the neighbouring junctions at time $t+1$ get affected by choosing the action a_t^j, at time t .

Each agent present at the junction obtains the cost feed back signal from its neighbours and updates its Q-factors by the Q-learning update rule using either ϵ-greedy or UCB based exploration strategy. Each agent then decides its actions based on the current learnt Q-factors. The next section describes the Q-learning algorithm along with its exploration strategy that is used by the agent to update its Q-factors and subsequently for choosing actions.

III. Algorithm

In this section, we describe the Q-learning algorithm that we use to update the Q-factors and thereby obtain suitable TSC policies.

The Q-factors or state-action value function $Q^\pi(s, a)$ for a policy π indicates how good an action a is in state s if we initially choose a in s and subsequently follow π [17]. The optimal state-action value function (optimal Q-factors) $Q^*(s, a)$ gives the minimum expected sum of discounted single-stage costs by choosing an action a in state s at time 0 and subsequently following the optimal policy $\pi^* = \{\mu^*, \mu^*, \ldots\}$ thereafter, i.e.,

$$Q^*(s, a) = k(s, a) + \mathbb{E}[\sum_{t=1}^{\infty} \alpha^t k(s_t, \mu^*(s_t)) | s_0 = s, a_0 = a, \mu^*], \ \forall (s, a), \tag{2}$$

where $k(s, a)$ is the immediate or the single-stage cost function in state s under action a. Once we determine

$Q^*(s,a)$ for all state-action pairs, we can obtain the optimal action for a state s by choosing the action that minimizes $Q^*(s,a)$, and hence

$$\mu^*(s) = \arg\min_{a \in A} Q^*(s,a), \quad \forall\ s. \tag{3}$$

Thus, in order to obtain the optimal policy, we need to obtain optimal Q-factors.

Q-learning is an iterative method that modifies the estimates of Q-factors, according to the Q-learning update rule so that, the estimates converge to the optimal Q-factors. In our setting, each agent follows the Q-learning update rule (4) to update its Q-factors. The agent at junction j follows the following update rule: Let $Q_0^j(s^j,a^j) = 0,\ \forall\ s^j \in S^j,\ a^j \in A. \quad \forall t \geq 0,$

$$Q_{t+1}^j(s^j,a^j) = Q_t^j(s^j,a^j) + \gamma(t)(c_j(t) + \alpha \min_{b \in A} Q_t^j(s^{j'},b) - Q_t^j(s^j,a^j)), \tag{4}$$

where s^j and $s^{j'}$ correspond to the segregated queue-length vectors at the junction j at times t and $t+1$, respectively, when we set the signal duration of the current phase as a^j at time t according to the ε-greedy exploration or through the UCB exploration strategy. The cost $c_j(t)$ is computed from the neighbouring junctions as mentioned in (1). The step sizes $\gamma(t),\ t \geq 0$ in (4) satisfy the requirement, $\gamma(t) > 0\ \forall t$ and further,

$$\sum_t \gamma(t) = \infty, \sum_t \gamma(t)^2 < \infty. \tag{5}$$

In the case of ε-greedy exploration strategy, with probability $1-\varepsilon$, we choose the action $a = \arg\min_c Q_t(s^j,c)$ and with probability ε, a is a random feasible action in state s^j. With a suitable choice of $\varepsilon > 0$, one may be able to strike a balance in the exploration versus exploitation trade off.

In the case of UCB-based exploration strategy, we compute the action a to be selected based on the upper confidence bound index. The index is constructed as a sum of two terms. The first term corresponds to the learnt Q-factor for the state-action pairs. The second term is inversely proportional to the number of times an action has been chosen in a given state. Note that the UCB algorithm setting for stateless MDP (the multi-armed bandit) is given in [3]. Here

$$a = \arg\max_{c \in A} \left\{ -Q_t^j(s^j,c) + \sqrt{\frac{\ln R_{s^j}(t)}{R_{s^j,c}(t)}} \right\}, \tag{6}$$

where $R_{s^j}(t)$ denotes the number of times the state s^j has been visited till time t and $R_{s^j,c}(t)$ denotes the number of times the action c has been chosen in state s^j until time t. The second term helps in exploration. If action c has not been chosen sufficient number of times in state s^j then the second term will dominate the first term and action c will be explored. As learning progresses, the first term dominates the second term and the action selection is based only on the Q-factors.

For a single agent (junction) setting, the Q-learning update rule (4) with both the exploration strategies converges to the optimal policy (see [18]). In the multi-agent setting when we construct the immediate cost based on the neighbouring junctions (as mentioned in (1)), the Q-learning update rule yields a policy that has significantly better performance over both the FST and the SAT algorithms (see Section IV).

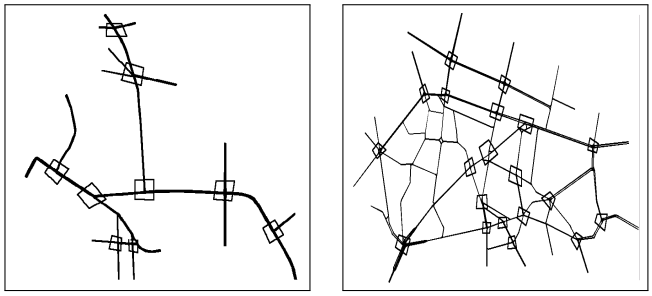
IV. Simulation results

We used the VISSIM traffic simulator developed by PTV-Tag for performance comparisons of the algorithms. VISSIM is a microscopic, time step and behaviour based simulation software developed to model urban traffic and public transit operations. The microscopic-modelling approach attempts to provide an accurate description of the traffic dynamics.

Experimental setup: We considered the following two different settings for the road networks:

- A nine-junction road network (9Jn) - a real road network around the Indian Institute of Science campus in Bangalore, India.
- A twenty-junction road network (20Jn)- a real road network in Bangalore, India.

TABLE I
DIFFERENT ROAD NETWORK LAYOUTS



(a) Nine-Junction Network (b) Twenty-Junction Network

Table I gives the two different road network layouts we considered in our experiments. In all our experiments, we set the discretized values for the actions to $\{low = 10s,\ medium = 20s,\ high = 30s\}$ and the results are averaged over five runs. The average number of vehicles entering the 9Jn and the 20Jn road network per hour was set to 5500 and 9650 respectively. In the plots shown, the X-axis corresponds to the iterations of the algorithm and the Y-axis corresponds either to the average delay experienced by the road users or the average stopped delay. Note that the average delay of the vehicles is computed as the average difference between (a) the time taken for a vehicle to travel from the source to the destination when there is no traffic and (b) that when there is traffic. Similarly, average stopped delay is computed as the average standstill time per vehicle near junctions.

In the FST algorithm, the time durations of the various phases have been chosen to be the best (in terms of average delay) over a wide range of fixed timing values. In the SAT algorithm, we have set the saturation level to 90% and the

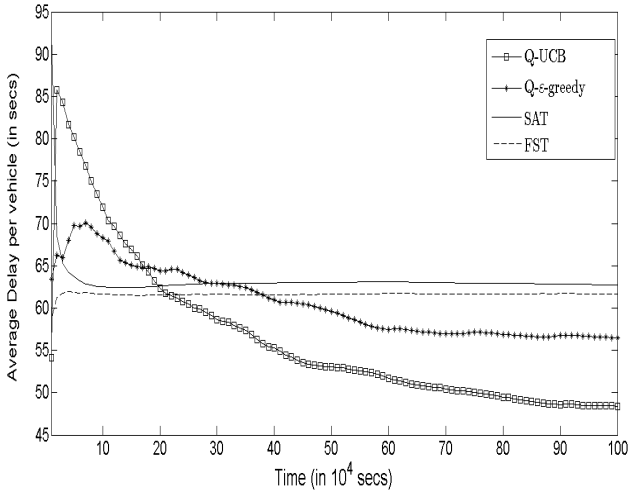


Fig. 1. Average delay performance comparison for the nine junction road network

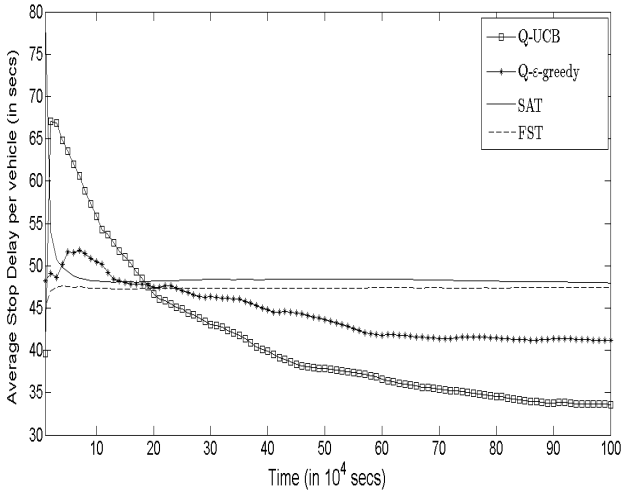


Fig. 3. Average stopped delay performance comparison for the nine junction road network

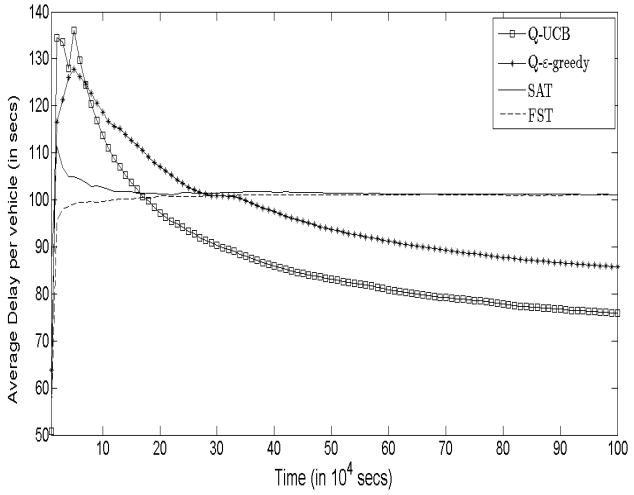


Fig. 2. Average delay performance comparison for the twenty junction road network

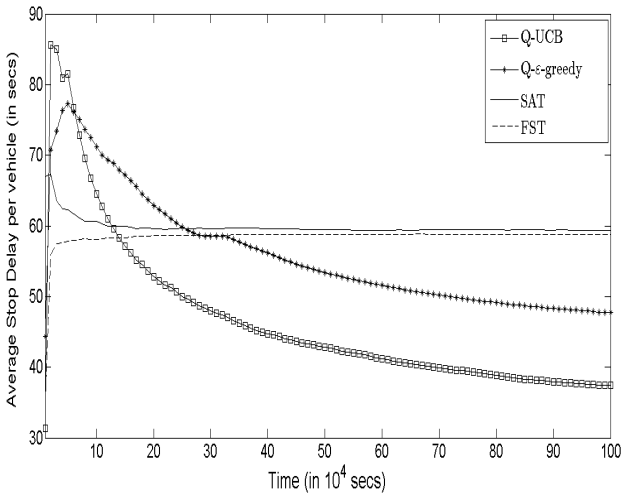


Fig. 4. Average stopped delay performance comparison for the twenty junction road network

minimum phase duration to 20 seconds. Also, we computed the (parameter) maximum cycle length as in [16] by using the factor parameter set to 2. Please refer [15] for a complete description of the SAT algorithm. The MARL algorithm with UCB-based exploration is denoted as Q-UCB and the MARL algorithm with ε-greedy exploration is denoted as Q-ε-greedy. The discount factor α was set to 0.9. We set $\epsilon = 0.1$ in the Q-ε-greedy algorithm. We set the step-size $\gamma = 0.1$ for the first 10^5 seconds and then gradually decrease it as $\frac{C}{n}$, for $n > 10^5$ seconds where $C = 10^4$. Note that the step-sizes chosen satisfy (5).

From Figs. 1 and 2 , it can be seen that our algorithms exhibit significantly better performance over the FST and SAT algorithms in terms of average delay for the nine and twenty junction road networks. In the initial stages of learning, our algorithms have higher average delay compared to the FST and SAT algorithms. However, as learning progresses, the average delay of the vehicles is greatly reduced after 3×10^5 seconds. Figs. 3 and 4 demonstrate the effectiveness of the dynamic TSC policies obtained by our MARL algorithms in terms of the average stopped delay. The fixed timing algorithm due to its static nature causes a large delay for the road users. The SAT algorithm despite being dynamic in nature is unable to adapt quickly to the underlying traffic. Our algorithms, on the other hand, by adapting to the traffic allocate green time dynamically to phases in an efficient way so that the delay is minimized. Q-UCB is seen to perform even better than Q-ε-greedy owing to the fast nature of exploration.

The final performance statistics for all the algorithms in terms of the average delay time, average number of stops and average delay has been summarized in Table II. Here,

the entries in the first column correspond to the road setting as well as the algorithm used. For instance, 9JnQ-UCB corresponds to the 9Jn road network using the multi-agent Q-learning algorithm with UCB based exploration. It can be easily seen from the table that the performance indices in terms of average number of stops, average stopped delay and average delay, respectively, when using our algorithms show a significant improvement over the FST and the SAT algorithms. For instance, in the twenty junction scenario we get approximately 20 and 10 seconds reduction in average delay for Q-UCB and Q-ε-greedy compared to the FST and SAT algorithms. The dependence between junctions for the 20 junction road network is more compared to the 9 junction road network. Thus, our algorithms show better performance indices in the 20 junction scenario.

It is interesting to observe that the policies obtained from our algorithms are able to produce a self-organizing behaviour of the traffic lights. Initially, the starting phases in the RR schedule are chosen randomly. Using our MARL algorithms to update Q-factors asynchronously and deriving policies from the learnt Q-factors reduces the average delay considerably. As the goal is to minimize the global cost objective, i.e., the overall delay in the network, the decentralized nature of updating Q-factors and taking decisions results in a self-organizing behaviour of the traffic lights.

	Average delay [s]	Average stopped delay [s]	Average number of stops
9JnFST	47.40	61.72	2.14
9JnSAT	47.92	62.74	2.11
9JnQ-ε-greedy	41.13	56.46	1.82
9JnQ-UCB	33.60	48.41	1.63
20JnFST	58.78	101.06	4.03
20JnSAT	59.30	101.10	4.01
20JnQ-ε-greedy	47.61	85.74	3.56
20JnQ-UCB	37.31	75.82	3.04

TABLE II

AVERAGE PERFORMANCE STATISTICS COMPARISONS FOR THE TSC ALGORITHMS FOR TWO ROAD NETWORK LAYOUTS

V. CONCLUSION

We formulated the traffic signal control problem as a discounted cost MDP and applied the Q-UCB and Q-ϵ-greedy algorithms. Our algorithms are seen to converge to policies that significantly outperform the FST and SAT algorithms. Q-UCB even outperforms Q-ϵ-greedy as the exploratory action selection for Q-UCB is based on both the learnt Q-values as well as the number of times the action has been chosen in the past, whereas Q-ϵ-greedy's action selection is based only on the learnt Q-values. Our algorithms are adaptive in nature and do not require model information on the transition probabilities. One could extend our algorithms to include eligibility traces [17] for faster convergence. Also, the discretization used for the queue-length and the time-duration of the phases has been fixed in our experiments. One could tune these parameters adaptively for optimal performance as in [14]. In the current work we assumed zero delay in the message movement between the junctions for constructing the cost function. But, in a real setting there will be inherent delays. Thus, in the future, one could develop MARL algorithms accounting for these practical difficulties. While we did not prove convergence of Q-learning in the muli-agent setting, we intend to do this in the future.

REFERENCES

[1] M. Abdoos, N. Mozayani, and A.L.C. Bazzan. Traffic light control in non-stationary environments based on multi agent q-learning. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 1580–1585. IEEE, 2011.

[2] I. Arel, C. Liu, T. Urbanik, and A.G. Kohls. Reinforcement learning-based multi-agent system for network traffic signal control. *Intelligent Transport Systems, IET*, 4(2):128–135, 2010.

[3] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[4] B. Bakker, S. Whiteson, L. Kester, and F. Groen. Traffic light control by multiagent reinforcement learning systems. *Interactive Collaborative Information Systems*, pages 475–510, 2010.

[5] A.L.C. Bazzan. Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. *Autonomous Agents and Multi-Agent Systems*, 18(3):342–375, 2009.

[6] C. Cai, C.K. Wong, and B.G. Heydecker. Adaptive traffic signal control using approximate dynamic programming. *Transportation Research Part C: Emerging Technologies*, 17(5):456–474, 2009.

[7] S.B. Cools, C. Gershenson, and B. D' Hooghe. Self-organizing traffic lights: A realistic simulation. *Advances in Applied Self-organizing Systems*, pages 41–50, 2008.

[8] Y. Dai, D. Zhao, and J. Yi. A comparative study of urban traffic signal control with reinforcement learning and adaptive dynamic programming. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1–7. IEEE, 2010.

[9] S. El-Tantawy and B. Abdulhai. An agent-based learning towards decentralized and coordinated traffic signal control. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 665–670. IEEE, 2010.

[10] L. Kuyer, S. Whiteson, B. Bakker, and N. Vlassis. Multiagent reinforcement learning for urban traffic control using coordination graphs. *Machine Learning and Knowledge Discovery in Databases*, pages 656–671, 2008.

[11] J.C. Medina and R.F. Benekohal. Traffic signal control using reinforcement learning and the max-plus algorithm as a coordinating strategy. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 596–601. IEEE, 2012.

[12] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Yibing Wang. Review of road traffic control strategies. *Proceedings of the IEEE*, 91(12):2043–2067, 2003.

[13] L.A. Prashanth and S. Bhatnagar. Reinforcement learning with function approximation for traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 12(2):412–421, 2011.

[14] L.A. Prashanth and S. Bhatnagar. Threshold tuning using stochastic optimization for graded signal control. *Vehicular Technology, IEEE Transactions on*, 61(9):3865–3880, 2012.

[15] S Richter. Learning traffic control-towards practical traffic control using policy gradients. *Albert-Ludwigs-Universitat Freiburg, Tech. Rep*, 2006.

[16] A. Salkham, R. Cunningham, A. Garg, and V. Cahill. A collaborative reinforcement learning approach to urban traffic control optimization. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on*, volume 2, pages 560–566. IEEE, 2008.

[17] R.S. Sutton and A.G. Barto. *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press, 1998.

[18] C.J.C.H. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.

[19] Y. Yin. Robust optimal traffic signal timing. *Transportation Research Part B: Methodological*, 42(10):911–924, 2008.