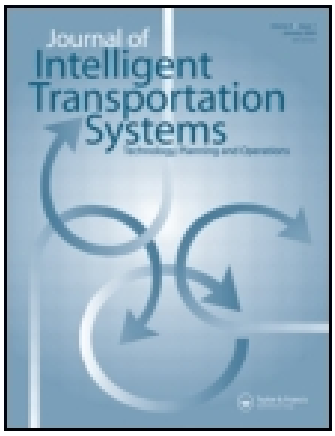


Journal of Intelligent Transportation Systems: Technology, Planning, and Operations
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/gits20

Design of Reinforcement Learning Parameters for Seamless Application of Adaptive Traffic Signal Control

Samah El-Tantawy[ab], Baher Abdulhai[a] & Hossam Abdelgawad[ac]
[a] Civil Engineering Department, University of Toronto, Toronto, Ontario, Canada
[b] Engineering Mathematics Department, Cairo University, Giza, Egypt
[c] Department of Civil Engineering, Faculty of Engineering, Cairo University, Giza, Egypt
Accepted author version posted online: 14 Jun 2013.Published online: 23 May 2014.

To cite this article: Samah El-Tantawy, Baher Abdulhai & Hossam Abdelgawad (2014) Design of Reinforcement Learning Parameters for Seamless Application of Adaptive Traffic Signal Control, Journal of Intelligent Transportation Systems: Technology, Planning, and Operations, 18:3, 227-245, DOI: 10.1080/15472450.2013.810991

To link to this article: http://dx.doi.org/10.1080/15472450.2013.810991

Journal of Intelligent Transportation Systems, 18:227–245, 2014

ISSN: 1547-2450 print / 1547-2442 online
DOI: 10.1080/15472450.2013.810991

Taylor & Francis
Taylor & Francis Group

Design of Reinforcement Learning Parameters for Seamless Application of Adaptive Traffic Signal Control

SAMAH EL-TANTAWY,[1,2] BAHER ABDULHAI,[1] and HOSSAM ABDELGAWAD[1,3]

[1]Civil Engineering Department, University of Toronto, Toronto, Ontario, Canada
[2]Engineering Mathematics Department, Cairo University, Giza, Egypt
[3]Department of Civil Engineering, Faculty of Engineering, Cairo University, Giza, Egypt

Adaptive traffic signal control (ATSC) is a promising technique to alleviate traffic congestion. This article focuses on the development of an adaptive traffic signal control system using Reinforcement Learning (RL) as one of the efficient approaches to solve such stochastic closed loop optimal control problem. A generic RL control engine is developed and applied to a multi-phase traffic signal at an isolated intersection in Downtown Toronto in a simulation environment. Paramics, a microscopic simulation platform, is used to train and evaluate the adaptive traffic control system. This article investigates the following dimensions of the control problem: 1) RL learning methods, 2) traffic state representations, 3) action selection methods, 4) traffic signal phasing schemes, 5) reward definitions, and 6) variability of flow arrivals to the intersection. The system was tested on three networks (i.e., small, medium, large-scale) to ensure seamless transferability of the system design and results. The RL controller is benchmarked against optimized pretimed control and actuated control. The RL-based controller saves 48% average vehicle delay when compared to optimized pretimed controller and fully-actuated controller. In addition, the effect of the best design of RL-based ATSC system is tested on a large-scale application of 59 intersections in downtown Toronto and the results are compared versus the base case scenario of signal control systems in the field which are mix of pretimed and actuated controllers. The RL-based ATSC results in the following savings: average delay (27%), queue length (28%), and l CO_2 emission factors (28%).

Keywords Adaptive Traffic Signal Control; Reinforcement Learning; Temporal Difference Learning

INTRODUCTION

Population is steadily increasing worldwide. Consequently, the demand for mobility is increasing, traffic congestion situations are deteriorating, and undesirable changes in the environment are becoming major concerns. Until relatively recently, infrastructure improvements have been the primarily method to cope with congestion. However, tight constraints on financial resources and physical space have accentuated the consideration of a wider range of options. Therefore, the emphasis has shifted to improving the existing infrastructure by optimizing the utilization of the available capacity through intelligent transportation systems (ITS). Adaptive traffic signal control (ATSC) has the potential to significantly alleviate traffic congestion as opposed to the commonly used pretimed and actuated control systems for isolated intersections (McShane, Roess, & Prassas, 1998).

Although existing adaptive traffic signal control systems (e.g., SCOOT: Hunt, Robertson, Bretherton, & Winton, 1998; SCATS: Sims & Dobinson, 1979; PRODYN: Farges, Henry, & Tufal, 1983; OPAC: Gartner, 1983; RHODES: Head, Mirchandani, & Sheppard, 1992) offer a wide range of performance improvements over pretimed and actuated signal control, they still suffer from the following limitations (Abdulhai & Kattan, 2003; Bazzan, 2009):

- Curse of dimensionality when handling several intersections simultaneously.

Address correspondence to Samah El-Tantawy, Civil Engineering Department, University of Toronto, Toronto, Ontario, Canada. E-mail: samah.el.tantawy@utoronto.ca

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/gits.

- Absence of an accurate traffic modeling framework that captures the dynamic and stochastic traffic assignment in response to signal control optimization changes.
- Reliance on frequently malfunctioning magnetic loop detectors for detection and estimation.
- Employing centralized control systems (in SCOOT [Hunt et al., 1998] and SCATS [Sims & Dobinson, 1979]) that are not scalable or robust, especially in cases of communication failure between the intersections and the traffic management center.
- Highly skilled labor requirements, which are often hard to maintain even at large Cities with ample resources.

Self-learning traffic control systems are emerging adaptive traffic control methods that rely on learning the control policy from the direct interaction with the traffic environment without the need of a predefined model for the environment and also without the need of human intervention.

Reinforcement learning (RL) is a machine learning algorithm that has shown great potential for self-learning traffic signal control in the stochastic traffic environment (Abdulhai & Kattan, 2003; Bazzan, 2009; Chen & Cheng, 2010; El-Tantawy & Abdulhai, 2010). The core difference between RL and the supervised learning is that correct input/output pairs are not available apriori; instead the learning is conducted to model interaction between the learner/agent and its environment. The basic concept of RL is concerned with an agent (e.g., traffic light) interacting with its environment (e.g., traffic network) that is modeled as a Markov decision process (MDP) in which the agent acts as the controller of the process (Sutton & Barto, 1998). The agent iteratively observes the state of the environment, takes an action accordingly, receives a feedback reward for the actions taken, and adjusts the policy until it converges to the optimal mapping from states to actions. The mapping from states to actions is also known as the control policy. In RL, the control agent interacts with the environment to learn the optimal policy, offering a closed-loop optimal control policy in which the control strategy (action) is determined based on the feedback measurements (state and reward) from the environment. Accumulating the maximum reward not only requires the traffic signal control agent to exploit (or reinforce) the best experienced actions, but also explores new actions to discover better actions selections in the future. To balance the exploration and exploitation in RL, action selection algorithms such as -greedy and softmax are typically used in the literature (Gosavi, 2003).

Numerous RL algorithms exist in the literature (Gosavi, 2003; Sutton & Barto, 1998); however, the temporal difference (TD) learning methods are the most relevant to the ATSC control problem, which can be viewed as a continuous (infinite-horizon) stochastic control problem. TD methods include TD (0) (e.g., Q-Learning, SARSA), and Eligibility Traces TD(λ) (e.g., Q(λ) and SARSA(λ)). The main challenge in designing any RL system is the design of the following elements/parameters: the learning method, state definition, action definition, reward definition, and action selection method. In the literature, different RL algorithms, and more specifically TD algorithms, have been investigated separately without providing quantitative justification for the RL-based system design in solving the traffic signal control problem.

This article presents a comprehensive investigation of key issues in RL-based ATSC for isolated intersections by offering a rigorous analysis on the effect of the following design parameters on the performance of the traffic network (e.g., average delay, queue lengths, number of stops, etc.): (1) learning method, (2) traffic state representation, (3) action selection method, (4) traffic signal phasing scheme, (5) reward definition, and (6) variability of flow arrivals to the intersection.

The agent is designed to learn offline through a simulation environment (such as the microsimulation model we employ in our experiments) before field implementation. After convergence to the optimal policy, the agent can either be deployed in the field—by mapping the measured state of the system to optimal control actions directly using the learned policy—or it can continue learning in the field, by starting from the learned policy. In both cases, no model of the traffic system is required, which makes the proposed approach more appealing for field deployment.

In fact, the experiments were tested on a simulated intersection in downtown Toronto (Front Street and Bay Street), using real traffic counts, to investigate the sensitivity of the RL parameters. The system is generically designed to easily model any phase sequence with a configurable number of phases. Although our ultimate goal is to develop a coordinated traffic control for large networks, this article represents the first step toward this goal by testing the algorithm with various parameters on one intersection. This strategy revamps the interpretation of the results, as we would rather solidify the methodology first then expand the scope of the problem by testing the algorithm with the best set of parameters on a two-dimensional network of 5 intersections and finally on a large-scale network of more than 50 intersections.

The remainder of the article is structured as follows: First the major challenges and gaps in the existing literature are summarized, then the proposed framework is presented while highlighting how it addresses the limitations/gaps in the literature, and finally, the experimental setup and results are discussed and concluded.

RL-BASED ADAPTIVE TRAFFIC SIGNAL CONTROL: THE STATE OF THE ART

This section focuses on the studies that considered RL for adaptive traffic signal control (Abdulhai, Pringle, & Karakoulas, 2003; Arel, Liu, Urbanik, & Kohls, 2010; Balaji, German, & Srinivasan, 2010; Camponogara & Kraus, 2003; De Oliveira et al., 2006; Richter, Aberdeen, & Yu, 2007; Salkham, Cunningham, Garg, & Cahill, 2008; Shoufeng, Ximin, & Shiqiang, 2008; Thorpe, 1997; Wiering, 2000). Table 1 is presented to compare

Table 1 Adaptive traffic signal control approaches using RL: Comparative literature survey.

Criterion study	TD learning method	RL elements			Number of phases	Time step	Action selection method
		State	Phasing sequence	Reward/penalty			
Thorpe (1997)	SARSA(λ)	Vehicle counts in the links leading to intersection	Fixed	(Penalty) the time required to release a fixed volume of traffic through a road network	2	1 sec	∈-greedy
Wiering (2001)	Model-based Q-Learning	Number and location of vehicles in the links leading to intersection	Variable	(Penalty) the total delay incurred between successive decision points	6	NA	∈-greedy
Abdulhai et al. (2001)	Q-Learning	Queue lengths in the links leading to intersection	Fixed	(Penalty) the total delay incurred between successive decision points	2	1 sec	softmax
Camponogara and Kraus Jr (2003)	Q-Learning	Number and location of vehicles on the links leading to intersection	Fixed	(Penalty) the number of vehicles waiting at intersection	2	NA	∈-greedy
Oliveira et al. (2006)	Similar to Q-Learning	Vehicle counts in the links leading to intersection	Fixed	(Penalty) the squared sum of the incoming links' queues	2	120 sec	∈-greedy
Oliveira et al. (2006)	Similar to Q-Learning	Vehicle counts in the links leading to intersection	Fixed	(Penalty) the squared sum of the incoming links' queues	2	120 sec	∈-reedy
Richter et al. (2007)	Modified Q-Learning	Current cycle length, current phase durations, and detectors status	Variable	(Penalty) the number of cars that entered intersection over the last time step	4	5 sec	softmax
Shoufeng et al. (2008)	Q-Learning	Total delay of the intersection	Fixed	The total delay of the intersection.	4	120 sec	∈-greedy
Salkham et al. (2008)	Modified Q-Learning	Vehicle counts in the links leading to intersection	Variable	(Reward) the difference between the number of vehicles that manage to clear the junction during the last time step and the number of vehicles that are still waiting	Depends on the junction	240 sec	softmax
Balaji et al. (2009)	Q-Learning	The change in the total queue length in the last time step	Fixed	(Penalty) The change in the total queue length in the last time step	Depends on the junction	500 s	∈-greedy
Arel et al. (2010)	Q-Learning	The relative delay at each lane leading to intersection	Variable	(Reward) the savings in the delay	8	1 sec	∈-greedy

and contrast these studies. The rows represent the study introduced to solve the problem, while the columns represent the comparison criteria. The criteria include TD learning method, definition of each RL element (state, action [phasing sequence], and reward), and action selection method. In Thorpe (1997), the SARSA algorithm was tested on a 4×4 grid network with three traffic loads ranging from 100 to 1000 vehicles, and the results showed that it outperformed the fixed timing plans by 29%. In Wiering (2000), a model-based RL for traffic control was investigated in which the location of each car represents the system state; which resulted in a complex system that its state space grows exponentially with the number of vehicles in the system. In Abdulhai et al. (2003), Q-Learning was tested on a simulated two-phase signalized intersection without turning flows and the results showed that it outperformed the pre-timed control scheme for variable traffic flows by 44%. In De Oliveira et al. (2006), the authors tested Q-Learning with multiple partial models of the environment and a cellular automaton-model is used to model drivers' behavior. A more complex RL approach is tested in Richter et al. (2007) that is based on four different RL methods while the following assumptions are considered to simplify the problem: All vehicles move at uniform speed, and interactions between cars are ignored within one road segment. A collaborative reinforcement learning-based controller was tested in Salkham et al. (2008) in which each agent decide the phase splits every two cycles and the state of each phase

is represented by either busy/not busy which lack capturing the dynamics of congestion. The Q-Learning algorithm with neural network for value function approximation was utilized in Arel et al. (2010) and tested using a discrete event environment in which traffic arrivals followed Poisson distribution.

From this survey it is important to highlight the following remarks: Although these studies contributed significantly to the literature, each study considered a certain RL design without providing quantitative justification for the selected parameters. Moreover, most of these studies have considered a simplified simulation environment (Abdulhai et al., 2003; Arel et al., 2010; Camponogara & Kraus, 2003; De Oliveira et al., 2006; Richter et al., 2007), and/or assumed hypothetical traffic flows (Abdulhai et al., 2003; Arel et al., 2010; Camponogara & Kraus, 2003; De Oliveira et al., 2006; Richter et al., 2007; Shoufeng et al., 2008; Thorpe, 1997) which does not necessarily mimic the reality. This article investigates the effect of the following design parameters to bridge this gap in the literature: (1) learning method (Q-Learning vs. SARSA vs. TD(λ)), (2) traffic state representation (queue length vs. queues and arrivals vs. delay), (3) action selection method ($\in$-greedy vs. softmax vs. $\in$-softmax), (4) traffic signal phasing scheme (variable vs. fixed), (5) reward definition (delay vs. cumulative delay vs. balancing queues), and (6) variability of flow arrivals to the intersection (uniform vs. variable arrival rates). In addition, these experiments are tested on a microscopic simulation environment that models individual vehicles' behavior in a real intersection with real traffic flows.

The contribution of this article is not introducing RL as a new traffic control methodology, which already exists in the literature already cited. The contribution in the article is in the side-by-side in-depth comparison of different RL algorithms and different learning sets of parameters, which takes RL signal control a step forward toward practical implementation.

RL-BASED ADAPTIVE TRAFFIC SIGNAL CONTROL

A generic RL software agent is developed using the Java programming language in a way so that different learning methods, state representations, phasing sequence, reward definition, and action selection strategies can be tested for any control task. The interaction between the agent and the traffic simulation environment is represented in Figure 1.

The input parameters to the RL-based ATSC system are combination of (1) general RL-related parameters, and (2) ATSC problem-specific related parameters. The agent component implements the control algorithm; the agent is the learner and the decision maker (or action selector) that interacts with the environment by first receiving the system's state and the reward and then selecting an action accordingly. A generic agent model is developed such that different learning methods, action selection strategies state representations, phasing sequence, and reward definition can be tested for any control task. The simulation environment component models the traffic environment. In this article, Paramics, a microscopic traffic simulator, is used to model traffic environment (Quadstone Paramics, 2012). Paramics models stochastic vehicle flow by employing speed regulations, car following, gap acceptance, and overtaking rules. Paramics provides three methods of traffic assignment that could be employed at different levels: "all-or-nothing" assignment, stochastic assignment, and dynamic feedback assignment. In this application a dynamic stochastic traffic assignment was used where (1) a random noise was added to the travel cost to account for the heterogeneity among drivers' perception of travel cost, and (2) a dynamic feedback interval was used to update route travel times for familiar drivers in the simulation. Paramics Application Programming Interface (API) functions were used to construct the state, execute the action, and calculate the reward for each signalized intersection.

Temporal Difference Learning Methods

Due to the stochasticity associated with the traffic flow approaching signalized intersections, the ATSC problem can been modeled as a stochastic control problem in which the goal is to obtain a policy (mapping between each intersection traffic state s and signal action a) that maximizes the expected cumulative discounted reward (e.g., delay saving) at each traffic state. The expected cumulative discounted reward at state s is defined as the value function of the stateV(s) or state–action pair Q(s,a), also called Q-value. The traffic signal control agent updates the estimate of the value functions over time until it converges to the optimal values, and consequently, the optimal policies.

The temporal difference is a class of RL methods that update the estimates of the state values immediately after visiting the state in which the value function is estimated based on the difference between temporally successive estimations, which is called TD error, denoted TD^k,

$$TD^k\left(s^{k-1}\right) = \left[r^k + \gamma V^{k-1}\left(s^k\right) - V^{k-1}\left(s^{k-1}\right)\right] \quad (1)$$

$$V^k\left(s^{k-1}\right) = V^{k-1}\left(s^{k-1}\right) + \alpha^k TD^k\left(s^{k-1}\right) \quad (2)$$

where α^k and γ are the step size (or learning rate) at time k and the discounted factor, respectively. At time k, the value of state s, $V^k(s^{k-1})$, is updated using the observed reward r^k and the estimate $V^{k-1}(s^k)$. Equation 1 represents the simplest form of TD known as TD(0). Two types of TD algorithms are presented in the literature, namely. TD(0) and eligibility traces TD(λ).

TD(0)

TD(0), also known as 1-step TD, resembles the online form of dynamic programming value iteration, in which the traffic signal agent looks ahead one step in time and updates the value functions based on the next reward r^k. TD(0) can represent either an on-policy or an off-policy algorithm, depending on the updating procedure for the value functions as follows.

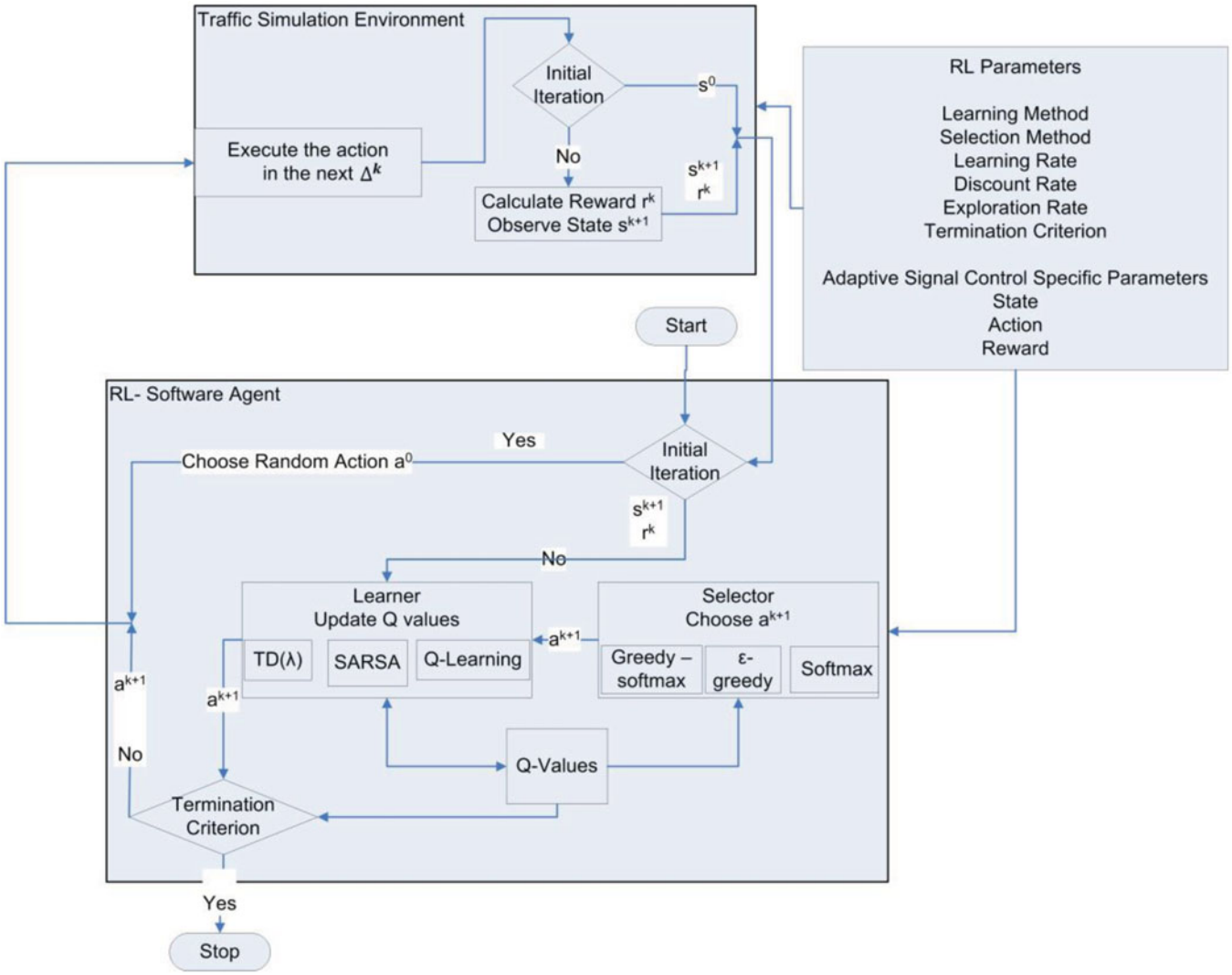


Figure 1 Agent–environment interaction.

Q-Learning: Off-Policy Algorithm

Q-Learning updates the estimated value functions (Q-values) using greedy actions (the action that has the highest value), independent of the policy being followed which involves exploratory actions (Sutton & Barto, 1998). Therefore, Q-Learning is an off-policy algorithm because the agent attempts to improve a policy while following another one's actions (see Eq. 3):

$$Q^k\left(s^{k-1}, a^{k-1}\right) = Q^{k-1}\left(s^{k-1}, a^{k-1}\right) + \alpha^k\left[r^k + \gamma \max_{a' \in A} Q^{k-1}\left(s^k, a^k\right) - Q^{k-1}\left(s^{k-1}, a^{k-1}\right)\right] \quad (3)$$

SARSA: On-Policy Algorithm

The Q-value functions are updated using actions (a') determined by a policy that involves exploratory actions (Sutton & Barto, 1998). SARSA algorithm is considered as an on-policy TD method since it is learning and improving the same policy that is followed in selecting the actions (see Eq. 4):

$$Q^k\left(s^{k-1}, a^{k-1}\right) = Q^{k-1}\left(s^{k-1}, a^{k-1}\right) + \alpha^k\left[r^k + \gamma Q^{k-1}\left(s^k, a^k\right) - Q^{k-1}\left(s^{k-1}, a^{k-1}\right)\right] \quad (4)$$

Eligibility Traces (TD(λ))

If the signal control agent looks ahead all the way into and until the end of the learning horizon (T) and updates the value functions based on all future rewards, this can be achieved through using eligibility traces algorithms (Sutton & Barto, 1998). An eligibility trace is an additional variable for each state that keeps

track of how often and how recently a state is visited. At each time step, the eligibility trace for each state is updated using Eq. 5:

$$e^k(s) = \begin{cases} \gamma\lambda e^{k-1}(s) & \text{if } s \neq s^k \\ \gamma\lambda e^{k-1}(s) + 1 & \text{if } s = s^k \end{cases} \quad (5)$$

where λ is referred as the trace-decay parameter. The update is then performed on the value estimates of all the states using Eq. 6:

$$TD^k\left(s^{k-1}\right) = \alpha^k\left[r^k + \gamma V^{k-1}\left(s^k\right) - V^{k-1}\left(s^{k-1}\right)\right] e^k\left(s^k\right) \quad (6)$$

Eligibility traces credit states for rewards obtained later. They allocate more credit for states visited recently than to states visited longer ago, which matches biological brain strategies for deciding on how recently received incentives should be used in combination with current incentives to determine actions (Jang, Sun, & Mizutani, 1997).

Applying the eligibility traces concept to the TD(0) methods (e.g., Q-Learning and SARSA) results in the algorithms Q(λ) and SARSA(λ) (Sutton & Barto, 1998). The design elements of the RL structure, that is, the definitions of state, action, reward, and action selection strategies for adaptive traffic signal control, are discussed next.

State Representation

The state is represented by a vector of $2 + N$ components, where N is the number of phases. The first two components are (1) index of the current green phase and (2) elapsed time of the current phase. Three RL models are developed for the remaining N components; each considers one of the state representations that are investigated separately in the literature as follows.

State Definition 1: Queue Length

State definition 1 is represented by a vector of N components that are the maximum queue lengths associated with each phase (Eq. 7), which is one of the most common state definitions in the RL-based ATSC literature (Abdulhai et al., 2003; Balaji et al., 2010; Richter et al., 2007):

$$s_i^k = \max_{l \in L_i} q_l^k \forall i \in \{1, 2, \ldots, N\} \quad (7)$$

where q_l^k is the number of queued vehicles in lane l at time k. The maximum queue is taken over all lanes that belong to the lane group corresponding to phase i, Li. Vehicle (v) is considered at a queue if its speed is below certain speed threshold(Sp^{Thr}), inthisstudySp^{Thr} = 7kph (Eq. 8):

$$q_l^k = \sum_{v \in V_l^k} q_v^k \quad (8)$$

$$< \overline{eq} > \text{ where } < eq > q_v^k = \begin{cases} 1 \text{ if } Sp_v^{k-1} > Sp^{Thr} \text{ and } Sp_v^k \leq Sp^{Thr} \\ -1 \text{ if } Sp_v^{k-1} \leq Sp^{Thr} \text{ and } Sp_v^k > Sp^{Thr} \\ 0 \text{ if } Sp_v^{k-1} \leq Sp^{Thr} \text{ and } Sp_v^k \leq Sp^{Thr} \end{cases} \quad (9)$$

where V_l^k is the set of vehicles travelling on lane l at time k.

State Definition 2: Arrival of Vehicles to the Current Green Phase and Queue Length at Red Phase

In this definition, one of the state vector components is the maximum arrivals in the green phase and the other $N - 1$ components are the maximum queue lengths for the red phases. A similar state definition is used in Thorpe (1997), De Oliveira et al. (2006), and Bingham (2001). This definition is represented by Eq. 9:

$$s_i^k = \begin{cases} \max_{l \in L_i} q_l^k \text{ if } i = \text{ red phase} \\ \max_{l \in L_i} Ar_l^k \text{ if } i = \text{ green phase} \end{cases} \quad (10)$$

$$\forall i \in \{1, 2, \ldots, N\}$$

where Ar_l^k is the number of arriving vehicles to lane l at time k.

A similar state definition is considered in the literature (Camponogara & Kraus, 2003; De Oliveira et al., 2006; Salkham et al., 2008; Thorpe, 1997; Wiering, 2000).

State Definition 3: Cumulative Delay

The vehicle cumulative delay is CD^v and Cd_v^k is the total time spent by vehicle v in a queue up to time step k. The cumulative delay for phase I is the summation of the cumulative delay of all the vehicles that are currently traveling on lane group Li (Eq. 9). Vehicles leave the system once they clear the stop line at the intersection. This state is also represented by N components where each component is the cumulative delay of the corresponding phase:

$$s_i^k = \sum_{l \in L_i} \sum_{v \in V_l^k} Cd_v^k \qquad \forall i \in \{1, 2, uN\} \quad (11)$$

$$\text{where } < eq > Cd_v^k = \begin{cases} Cd_v^{k-1} + \Delta^{k-1} \text{ if } Sp_v^k \leq Sp^{Thr} \\ Cd_v^{k-1} \text{ if } Sp_v^k > Sp^{Thr} \end{cases}$$

and where Δ^{k-1} is the previous time step before the decision point at time k and Sp_v^k is vehicle's speed at time k. A similar state definition is investigated in Shoufeng et al. (2008) and Arel et al. (2010).

State definitions 1 and 2 are indirect indicators of the cumulative delay encountered by vehicles at the intersection. It is

worthwhile to examine using the cumulative delay as a state representation directly as in state definition 3, although it is more difficult to measure in the field compared to queue lengths.

Action Definition

In this article, two phasing sequence schemes are investigated: fixed phasing sequence, in which a fixed order of phases is followed, and variable phasing sequence, in which the phasing sequence is not predetermined.

Action Definition 1: Fixed Phasing Sequence

The action is a binary number that indicates either extend (i.e., $a = 0$) or terminate the current green phase and move to the next phase (i.e., $a = 1$) (Eq. 11):

$$a^k = i, i \in \{0, 1\} \tag{12}$$

Action Definition 2: Variable Phasing Sequence

The action is the phase that should be in effect next (Eq. 12):

$$a^k = i, i \in \{1, 2, \ldots, N\} \tag{13}$$

It is worth noting that if the action is the same as the current green phase, then the green time for that phase will be extended by a specific time interval (1 sec). Otherwise, the green light will be switched to phase a after accounting for the yellow time (Y), all red (R), and the minimum green (G_{min}) times (Eq. 13):

$$\Delta^k = \begin{cases} G_{min}^{a^k} + Y^{a^k} + R^{a^k} & \text{if } a^k \neq a^{k-1} \\ 1 \text{ sec} & \text{if } a^k = a^{k-1} \end{cases} \tag{14}$$

Reward Definition

In this article, four definitions are considered for the immediate reward.

Reward Definition 1: Minimizing the Delay

A typical reward function considers the negative value of the delay experienced by the vehicles between two successive decision points (Eq. 14) (Abdulhai et al., 2003; Lu, Liu, & Dai, 2008; Shoufeng et al., 2008; Wiering, 2000). This typical definition, however, does not consider how long the vehicles were delayed from the time they approached the intersection:

$$r^k = -\sum_{i \in N} \sum_{l \in L_i} \sum_{v \in V_l^k} b_v^k . \Delta^{k-1} \tag{15}$$

$$b_v^k = \begin{cases} 1 & \text{if } Sp_v^k \leq Sp^{Thr} \\ 0 & \text{if } Sp_v^k > Sp^{Thr} \end{cases}.$$

Reward Definition 2: Maximizing the Reduction in the Total Cumulative Delay

The immediate reward is defined as the reduction (saving) in the total cumulative delay, that is, the difference between the total cumulative delays of two successive decision points (Arel et al., 2010). The total cumulative delay at time k is the summation of the cumulative delay, up to time k, of all the vehicles that are currently in the system. If the reward has a positive value, this means that the delay is reduced by this value after executing the selected action. However, a negative reward value indicates that the action results in an increase in the total cumulative delay (Eq. 15):

$$r^k = \sum_{i \in N} \sum_{l \in L_i} \left(\sum_{v \in V_l^k} Cd_v^k - \sum_{v \in V_l^{k-1}} Cd_v^{k-1} \right) \tag{16}$$

Reward Definition 3: Minimizing and Balancing Queue Length

Reducing the cumulative delay does not guarantee preventing queue spilling back and therefore affecting minor streets. A reward can be defined as the reduction in the sum of the squared maximum queues (Eq. 16). The objective is to minimize and equalize the queue lengths across different phases. This definition is used in Balaji et al. (2010), Camponogara and Kraus (2003), and De Oliveira et al. (2006):

$$r^k = \sum_{i \in N} \left(\max_{l \in L_i} q_l^k \right)^2 - \sum_{i \in N} \left(\max_{l \in L_i} q_l^{k-1} \right)^2 \tag{17}$$

Reward Definition 4: Minimizing Stops

Reducing the cumulative delay or balancing queue lengths does not guarantee minimizing the total number of stops. A reward can be defined as the negative of the number of stops that occurred between two successive decision points:

$$r^k = -\sum_{i \in N} \sum_{l \in L_i} \sum_{v \in V_l^k} St_v^k \tag{18}$$

$$St_v^k = \begin{cases} 1 & \text{if } Sp_v^{k-1} > Sp^{Thr} \text{ and } Sp_v^k \leq Sp^{Thr} \\ 0 & \text{if } Sp_v^k > Sp^{Thr} \end{cases}$$

Action Selection Strategy

While accumulating the maximum reward requires the traffic signal control agent to exploit (or reinforce) the best experienced actions, it also has to explore new actions to discover better selection of actions for the future. Exploration allows visiting a larger number of state–action pairs, which is the condition of convergence to optimal policy (i.e., visit each state–action pair,

theoretically, an infinite number of times; Watkins & Dayan, 1992). To balance the exploration and exploitation in RL, algorithms such as ∈-greedy and softmax are typically used in the literature (Sutton & Barto, 1998).

∈-Greedy

In each iteration, the ∈ -greedy method is used for action selection in which the agent selects the greedy action most of the time except for ∈ amount of time, when it selects a random action uniformly.

This research adopts the ∈-greedy exploration method with a gradually decreasing rate of exploration, so that at the beginning the agent mostly explores, as it does not know much about the environment yet; more exploitation takes place toward the end of the learning as the agent converges to the optimal policy. Gradual decreasing rate is required to help the agent covering the entire state space in less learning time.

The gradually decreasing rate of exploration can be represented by an exponentially decreasing function as follows: $\in = e^{-En}$, as suggested by Jacob (2005), where E is a constant and n is the iteration number (the "age" of the agent).

Softmax

One disadvantage of the ∈-greedy selection method is that it treats all exploratory actions equally, irrespective of the estimated value function of each action; this means that it is as likely to chose the worst action as it is to chose the next-to-best action.

To overcome this limitation that may adversely affect the real-life applications, particularly where the worst action can be drastic, exploration can be concentrated on the most promising actions in terms of their estimated values. Thus, the selection of action probabilities can be represented by a proportional function to the estimated values, which is referred as softmax action selection strategy (Sutton & Barto, 1998). The most common softmax methods use a Boltzman distribution and result in the probability of selecting action *a* at state *s,* illustrated by Eq. 18:

$$P_s(a) = \frac{e^{\frac{Q(s,a)}{\tau}}}{\sum_{b\in A} e^{\frac{Q(s,b)}{\tau}}} \tag{19}$$

where A is the set of possible actions. The parameter τ is called the temperature. When τ is high, each action will have approximately the same probability to be selected (more exploration). When τ is low, actions will be selected proportionally to their estimated payoff (more exploitation).

Mixed ∈-Greedy and Softmax (∈-Softmax)

Although softmax explores the actions more intelligently, it exploits less aggressively than ∈-greedy. Also by using the softmax selection method, the exploration rate does not depend only on the value of τ (like ∈ -greedy method), but also depends on the magnitude of the Q-Values, so it is harder to find a parameter setting for τ as cited in Sutton and Barto (1998): "Most people find it easier to set the ∈ parameter with confidence; setting τ requires knowledge of the likely action values and of powers of ∈." The choice rule suggested by Wahba (2008) follows a mixed ∈-greedy and softmax action-choice model, with a $1 - \in$ probability of exploitation and ∈ probability of exploration (Eq. 18):

$$P_s(a) = \begin{cases} \begin{cases} 1 & Q(s,a) = \max_{a'\in A} Q(s,a') \\ 0 & \text{otherwise} \end{cases} & \text{if exploiting} \\ \dfrac{e^{Q(s,a)/\tau}}{\sum_{b\in A} e^{Q(s,b)/\tau}} & \text{if exploring} \end{cases} \tag{20}$$

In this method, the greedy action is still given the highest selection probability, and all other actions are weighted according to their estimated values.

Learning Rate and Discount Rate

The step-size rule suggested by Gosavi (2003) has been adopted in this article as illustrated in Eq. 19:

$$\alpha^k = \frac{1}{v^k(s,a)} \tag{21}$$

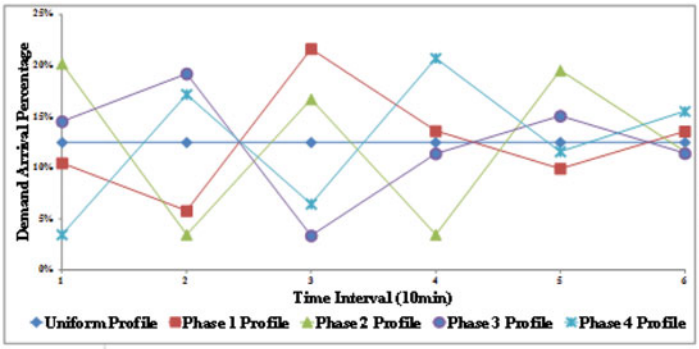
where $v^k(s,a)$ is the number of visits to a particular state–action pair (s,a). On the other hand, the discount rate $\gamma = 0.8$ is chosen.

EXPERIMENTAL SETUP AND RESULTS 1: ISOLATED INTERSECTION

The agent is tested on a major intersection (four approaches, three lanes, including an exclusive left turn lane) in downtown Toronto in the core of the financial district (Front Street and Bay Street; see Figure 2). This intersection is chosen as an example of an important multiphase intersection. The afternoon rush hour observed traffic demand data for the year 2005[1] are attached to the figure in a form of an origin–destination (OD) matrix. According to this demand, each of eastbound (E)/westbound (W) and northbound (N)/southbound (S) requires separate through and left-turn movements, resulting in four phases as shown in Figure 2.

A core element in the development of a microsimulation model is the calibration of the simulation model. Calibration entails refining the model to make sure traffic behaves without contradictions and refining model parameters to bring the model closer to observed conditions. During the initial network coding stage, network geometry was modeled and refined to a reasonable level of detail; however, once the demand was

[1]Data from years 2002, 2005, and 2009 were examined for the a.m. and p.m. rush hours. It is found that the p.m. rush hour for the year 2005 forms the worst-case scenario with regard to the total demand (arrival flows) approaching the intersection.



Traffic Volumes (Origin Destination Demand Matrix)

	E	S	N	W	Total
E	0	278	86	844	1208
S	88	0	721	71	880
N	188	806	0	100	1094
W	1223	121	134	0	1478
Total	1499	1205	941	1015	4660

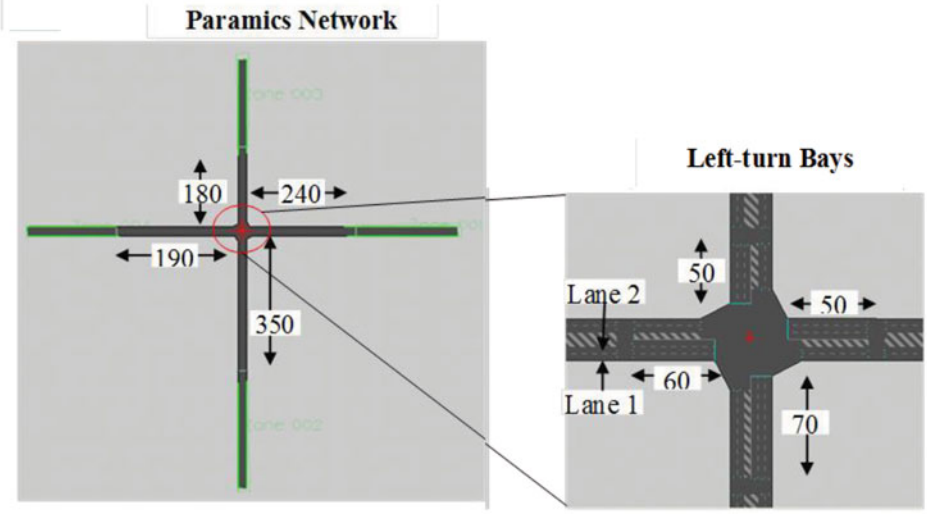


Figure 2 Small application—testbed network and demand profile.

generated by the OD estimation process and assigned to the network, further geometric model refinement was required. A set of system-wide microsimulation parameters was calibrated using a factorial design experiment procedure (high–low combination of parameters) by considering the default range of each parameter as detailed in Park and Qi (2004). These parameters are described next with their calibrated values:

- Headway (sec), the global mean target headway between a vehicle and a following vehicle. It is important to note that this headway is global target headway for all the vehicles in the model; this will not necessarily equal to the mean measured headway, as the relation between target and actual headways depends on traffic flow levels, driver behavior, and several other car following factors. Acceptable range 0.6–2.2 (calibrated value 0.75).
- Reaction time (sec), the mean global reaction time associated with the lag in time between a change in speed of the preceding vehicle and the following vehicle reaction to change. Acceptable range 0.3–1.9 (calibrated value 0.8).
- Time steps per second, number of time steps, previous to the current time step, that all vehicles record. This value is used to implement driver reaction time, by basing the change in speed of a following vehicle on the speed of the leading vehicle at a time in the recent past. Changing the size of the time steps per second allows the modeling of larger reaction times, or smaller time steps. Acceptable range 1–9 (calibrated value 2).
- Feedback interval (min), sets the period at which link times are updated into the routing calculation. Acceptable range 1—9 (calibrated value 3 min).
- Familiarity (%), percent of drivers aware of updated cost to destination each feedback interval. Acceptable range 0–100 (calibrated value 65%).
- Perturbation (%), models perception error or variation in perceiving true travel costs. Acceptable range 0–100 (calibrated value 10%).

Repetitions of the model were conducted for each combination of parameters and the results were assessed based on the GEH. It is noteworthy that the calibration effort considered the combination of the preceding values in tandem to minimize the differences between the observed flows and the simulation flows (i.e., minimizing GEH). The most significant parameters were found to be the time steps per second, headway factor, and reaction time. This process is repeated until the convergence criterion was met; in this case, a GEH of 3.85 was attained at the end of the calibration process.

The performance of the RL-based control agent is compared to two benchmarks: namely, optimal pretimed control and fully actuated control. The pretimed signal plan is optimized using the

Table 2 Small application—Green time splits using Webster method.

	NS		EW	
Parameter	Through	Left turn	Through	Left turn
Yellow	3	3	3	3
All red	2	2	2	2
Lost time	2	2	2	2
Demand per lane	463	188	473.6	278
Subtotal	413.3	71	684.1	134
Left turn Factors	1	1.05	1	1.05
Critical volume	463	197.4	684.1	291.9
v/S	0.24368	0.10389	0.36005	0.15363
C				
Total Ge				
Ge	32	14	47	20
Ga	27	9	42	15

Webster (1958) method, which resulted in cycle length of 120 sec. The fixed-time signal plan is optimized using the Webster method (Webster, 1958). The optimized effective green time for each phase is illustrated in

Table 2 using these parameters: saturation flow S = 1900 veh/greenhr/lane, yellow time (Y) = 3 sec, and all red time (R) = 2 sec.

A typical National Electrical Manufacturers Association (NEMA) actuated signal control system is implemented in which two loop detectors are embedded in each approach: stopline detector and upstream extension detector. The control logic for the NEMA controller is based on three timers to control the green phase as discussed next: minimum green, vehicle extension, and maximum green.

- Minimum green: green period that the selected phase must serve, typically governed by the minimum green time for pedestrian crossing to ensure the safe passage of pedestrians as explained later.
- Extension time (passage time): amount of green time extended by the controller during a green interval when a vehicle is detected. The passage time is applied to a vehicle travelling from the extension detector to the stopline during the green interval. The passage time only comes into effect once the minimum green has elapsed. It is calculated based on a speed of 40 kph and the distance between the stopline detector and the extension detector (∼45 m) with a typical range of 3–5 sec. For the testbed isolated intersection the passage time was found to be 4 sec.
- Maximum green: maximum green period for the selected phase. The maximum green time starts as soon as a conflicting vehicle is detected, during which the selected phase is green. The maximum green for the testbed isolated intersection was set to be the calculated Webster green time for the corresponding phase.

The RL platform interacts with Paramics through the Application Programming Interface (API) functions in Paramics (see Figure 1). While it is practically infeasible to continue the learning process indefinitely, a stopping criterion is specified to bring the RL to an end. In this implementation, the learning process is terminated after one hundred 1-hour simulation runs. For each demand data, two demand profiles (i.e., the temporal arrival percentages) are considered: constant (uniform) profile in which the demand is uniformly spread across the simulation horizon, and variable profile in which each movement has different randomized arrival rates around its mean arrival rate. In this experiment, we investigate the following parameters: (1) RL Method, (2) State Representation, (3) Action Definition, (4) Reward Definition, and (5) Action Selection Method, as shown in Table 3. The experiment considers different RL methods (column 2) that are discussed next. State rep. (column 3) takes a number from 1 to 3, which refers to the state representation definitions discussed in the second following subsection. Similarly, the experiments considered the action definitions (column 4), reward definitions (column 5), and action selection methods (column 6) that are discussed in subsequent subsections.

Effect of TD Methods on Convergence and Solution Quality (Exp. No. 1 to 4)

TD(0)

Q-Learning (Q(0)) and SARSA (SARSA(0)) are tested with the $\in$-greedy action selection method where $\varepsilon = e^{-En}$, E = 0.05. As shown in Figure 3a, both TD(0) algorithms converge to almost the same optimal policy (i.e. same optimal average delay). However, Q-Learning converges slightly more quickly than SARSA does. Early convergence of the Q-Learning agent is attributed to the fact that the agent is trained using an off-policy method, in which the agent learns actions that are not necessarily performed during the learning process. SARSA, on the other hand, takes the action selection into account and learns better policy than Q(0) but in a longer time. This is primarily because the SARSA agent visits more state–action pairs than Q-Learning over the same learning time, so it discovers the policy for those corresponding states. It is worth noting that in the early

Table 3 Small application—Design of experiments.

Experiment number	RL method	State rep.	Action	Reward	Action selection method
1	Q(0)	1	2	2	1
2	SARSA(0)	1	2	2	1
3	SARSA(0.5)	1	2	2	1
4	SARSA(1)	1	2	2	1
5	Q(0)	2	2	2	1
6	Q(0)	3	2	2	1
7	Q(0)	1	1	2	1
8	Q(0)	2	2	1	1
9	Q(0)	2	2	3	1
10	Q(0)	2	2	4	1
11	Q(0)	2	2	2	2
12	Q(0)	2	2	2	3

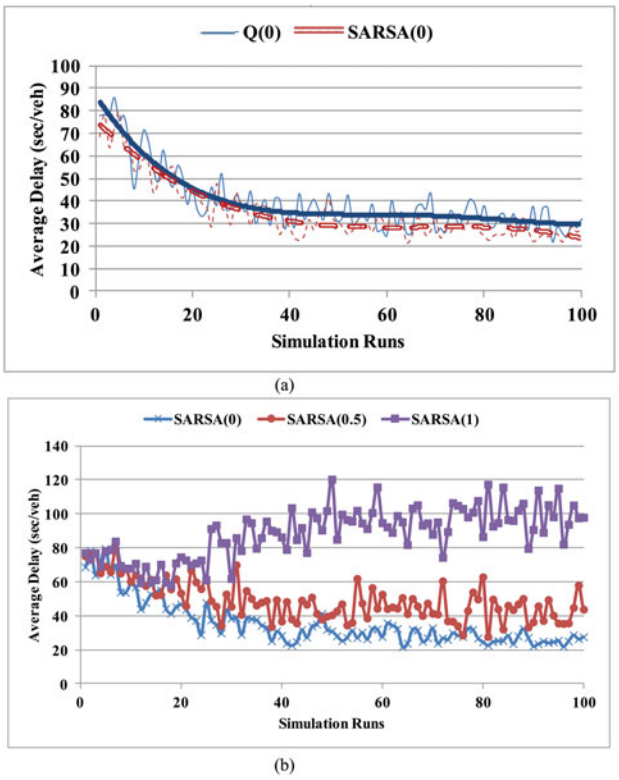


Figure 3 Small application—average delay using TD methods.

stages of the learning process, Q-Learning takes a longer time per iteration due to the search for the greedy action, compared to SARSA.

Eligibility Traces TD(λ)

Eligibility traces updates value functions based on all future rewards by allocating more credit for states visited recently than those states visited longer ago which should result in faster learning. Figure 3b illustrates three experiments for SARSA(λ) each with different λ: 0, 0.5, and 1. Unexpectedly, as shown in Figure 3b, the higher the value of λ, the worse the performance is. One possible explanation could be due to the nature of the signal control problem since the control task is a continuing task (i.e., not a finite episode) with a discounted reward in which looking ahead to future steps is less important compared to a finite episodic task with undiscounted reward. To the best of the authors' knowledge, no study in the literature investigated the effect of TD(λ) methods for such problems; however, most of the studies in the literature (de Queiroz, de Berrdo, & de Pádua Braga, 2006; Leng, Fyfe, & Jain, 2009) compared TD(λ) algorithms for finite episodic tasks in which there is an initial state and the target is to reach to a final state with an undiscounted reward. Also, the significant effect of TD(λ) appears particularly in problems where rewards are delayed by many steps (Kaelbling, Littman, & Moore, 1996), which is not the case in our problem since the reward definition is the difference in the cumulative delay between two successive decision points.

Effect of State Representation (Exp. No. 1, 5, and 6)

Three state representations are investigated under the variable demand profile, which forms a more challenging problem than the uniform arrival case. As shown in Table 4, no significant difference is reported between the performance of state definitions 1 and 2 in terms of the average delay; however, state definition 2 results in a lower average queue length and a lower standard deviation. By examining the traffic flow approaching the intersection, the EW direction carries higher demand than the NS direction, and both directions have the same minimum green values. State definition 2 drives the agent to extend the green for the EW phases as long as the number of arrivals is high and, at the same time, queue in the opposite direction is low; hence, it causes a minor increase in the delay compared to state definition 1 but endeavors more to equalize and reduce the average queue lengths.

The motivation for choosing state definition 3 is the direct relation between the state definition (cumulative delays) and the reward (reduction in the total cumulative delay). However, the experimental results (see Table 3) show that state definition 2 performs better. We suspect that the discretization bins for the state values could cause such performance. Determining the discretization bins for the queue length or the number of arrivals (i.e., for state definition 1 and 2) is more straightforward because its range is limited by the link lengths, while discretizing delay values can be more tricky due to the wide range of delay values.

It is found that considering the queue lengths at the red phases and the arrivals to the green phase gives the best state representations, especially in case of unbalanced demand across directions.

Effect of Action Definition (Exp. No. 1 and 7)

As shown in Table 4, fixed and variable phasing sequence schemes are tested under both uniform and variable profiles. Interestingly, under the uniform demand arrivals case, the fixed phasing sequence scheme has slightly better performance than the variable phasing sequence, in terms of the average delay and the percentage of stopped vehicles case. Our interpretation is that in the fixed phasing sequence case, it is easier for the agent to find the optimal policy, because under uniform demand arrival rates there is less need to jump phases. Thus, activating the green phases in order will result in good performance. Also, given that the agent has only two actions to choose from, the agent can easily learn when to extend the green based on the observed traffic state. On the other hand, variable phasing sequence performs better under variable demand profile (saving 35% in average delay). Given that these results were obtained under a demand level that is considerably high, it is anticipated that the agent will obtain much better results using a variable phasing sequence in cases of lower demand and variable arrivals profile. In such highly stochastic arrival patterns with low demand,

Table 4 Small application—Experimental results.

	Demand profile	Average delay (sec)	Throughput (veh)	Percent of stopped vehicles	Average stops	Average queue length per approach	Standard deviation of queue lengths
State							
1	Variable	30.7	4343	82	1.7	21.1	3.5
2		31.9	4354	81.2	1.7	16.2	0.5
3		43.3	4324	86.9	2.2	22.7	2.2
Action							
1	Uniform	25.2	4388	78.9	1.3	12.4	5.3
2		29.6	4380	80.4	1.4	14.6	0.04
1	Variable	47.1	4305	86.8	2.1	25.6	2.7
2		30.7	4343	82	1.7	21.1	3.5
Reward							
1	Variable	50.4	4152	84.8	2.1	25.2	4.2
2		30.7	4343	82	1.7	21.1	3.5
3		41.1	4297	84.8	2.1	21.6	5.6
4		193	2296	76.7	2.6	63.7	7
Pretimed							
Uniform	48.9	4317	81.1	1.6	26.1	6	
Variable	62.2	4209	88.9	2.1	33.9	1.8	
Improvement using RL (%)	Uniform	39.5	1.5	0.9	12.5	44.1	99.3
Variable	48.7	3.4	8.7	19.0	52.2	72.2	
Actuated							
Uniform	44.4	4267	80.6	1.6	23.2	5	
Variable	60.6	4150	83.8	1.9	30.4	2	
Improvement using RL (%)							
Uniform	33.33	2.65	0.25	12.50	37.07	99.20	
Variable	49.34	4.65	2.15	10.53	30.59	75.00	

the current traffic state is more affected by the arrival rate in current interval than the queue spill-back from the last interval, which is the case in the high-demand case. It is noteworthy that the RL-based control with variable phasing sequence results in the same average delay regardless of the arrival profiles, which indicates the robustness of the RL controller.

Effect of Reward Definition (Exp. No. 1, 8 to 10)

Four reward definitions (Eqs. 14, 15, 16, and 17) are examined under variable demand profile. As shown in Table 4, reward definition 1 performs worse than reward definition 2 in terms of the average delay and also the throughput. This finding was expected since reward definition 1 does not reflect the cumulative delay for vehicles, so the agent finds it better to activate the green for the highest queue that would minimize the delay between the two successive decision points regardless of the delay experienced by the queued vehicles before the last decision point. Hence, the agent learned to activate the green for through movements (phase 1 and 3) more than the left-turn movements (phase 2 and 4), which maximize its objective; however, taking this action negatively affects the total intersection delay and consequently the average delay. This reward definition suits better intersections with two phases as conducted in Abdulhai et al. (2003) and Lu et al. (2008).

Considering the reward as the reduction in the delay (reward definition 2) allows the agent to learn the optimal policy that results in the lowest average delay. Although the ultimate goal is to reduce the cumulative delay, the average numbers of stops and the average queue lengths are also decreased. However, to practically assess this reward definition, the availability of an advanced video tracking processing system or GPS devices in all vehicles can be an issue.

Although considering reward definition 3 reduced the average queue length similar to reward definition 2, the reduction in the average delay is not significant. This could be due to the unbalanced demand of the EW and NS, in which the signal will allocated more time to the EW phases in order to balance and minimize the queue length and hence increase the delay of the vehicles in NS direction. It is found that considering reward definition 3 (i.e., minimize both the queue lengths and the average delay) suits better intersections with balanced demand in all approaches as conducted in De Oliveira et al. (2006); otherwise, it will only minimizes the average queue length with high average delay.

Considering the objective as minimizing the number of stops results in a minimum percentage of stopped vehicles, but increases the average delay and queue length. This is due to the fact that the agent prefers to extend the green time as much as possible for the through phases while giving the minimum green only for the left-turn phases to minimize the number of stops between two successive decision points, which leads to decreasing the throughput and increasing the cumulative delay and the queue lengths for the other approaches. Ironically, it is found that the number of stops per vehicle increased. By

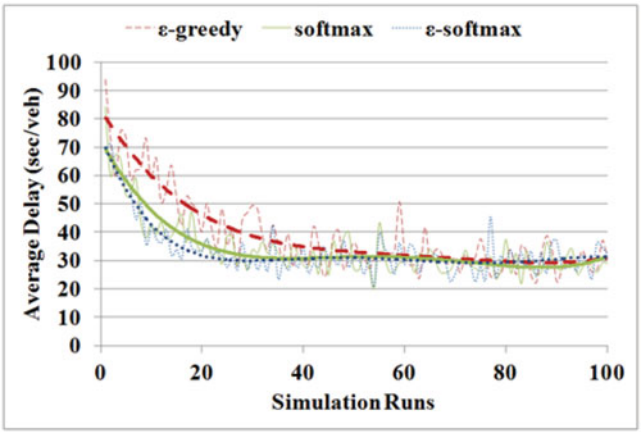


Figure 4 Small application—average delay for different action selection methods.

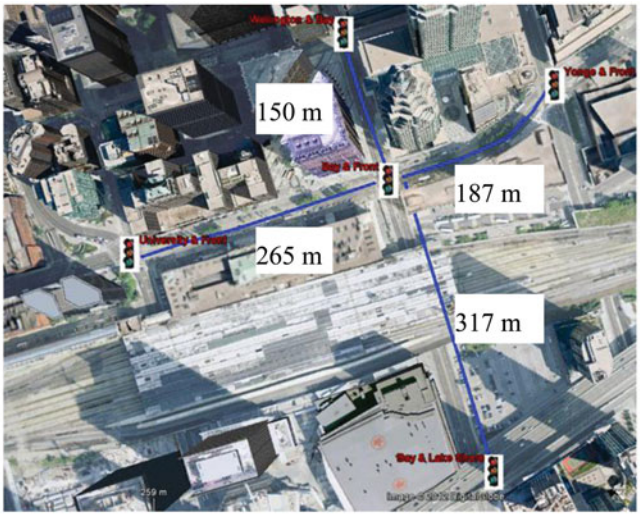
visualizing the simulation and examining the intersection geometry, it was found that the queue at lane 2 (Figure 2) is associated with vehicles willing to either continue through or make a left turn, and if many vehicles are willing to make a left turn, they will accordingly block lane 2. Therefore, the infrequent activation of the left-turn phases results in stopping each left-turn vehicle many times before it leaves the intersection and hence increasing the number of stops per vehicle. It is recommended therefore to use this reward definition in intersections with two phases with low left-turn demand.

Effect of Action Selection Method (Exp. No. 1, 11 and 12)

A set of experiments is adopted to compare two $\in = e^{-En}$ $\in$-greedy exploration methods: (1) a constant exploration rate ($\in$), and (2) a gradually exponentially decreasing exploration rate ($\in = e^{-En}$), as suggested by Jacob (2005), where E is a constant and n is the iteration number (the "age" of the agent). The gradual decrease of the exploration rate is found to be more promising, as many state–action pairs are visited during the learning stage/process. A value of $E = 0.05$ enables the agent to achieve the compromise of high convergence speed while discovering a large state–action space. Similar experiments are conducted to find the best τ and the same conclusions are drawn; therefore, $\tau = e^{-En}$, $E = 0.05$ are used in the conducted experiments.

The learning curve for $\in$-greedy, softmax, and $\in$-softmax are illustrated in Figure 4. As shown in Figure 4, all action selection methods eventually converged to the same optimal policy and optimal average delay. However, softmax converges faster than $\in$-greedy. The $\in$-softmax, on the other hand, synergizes the former two methods and therefore, converges faster than each of them. Also, the $\in$-softmax manifests a good learning start point; therefore, it is recommended to use $\in$-softmax for the learning in the field.

Table 4 compares the performance of the RL-based controller to two benchmarks: namely, pretimed control and fully actuated control with variable phasing sequence (refer to last eight rows in Table 3). As shown in Table 4, the RL agent outperforms both the pretimed and the actuated control regardless of the demand profile. Since the traffic flow is considerably high in all approaches, no significant difference is reported between the pretimed and the actuated control, especially in the case of variable profiles. However, the RL-based approach exhibits significant savings in average delay, stops, and queue lengths, especially in the case of variable profiles.



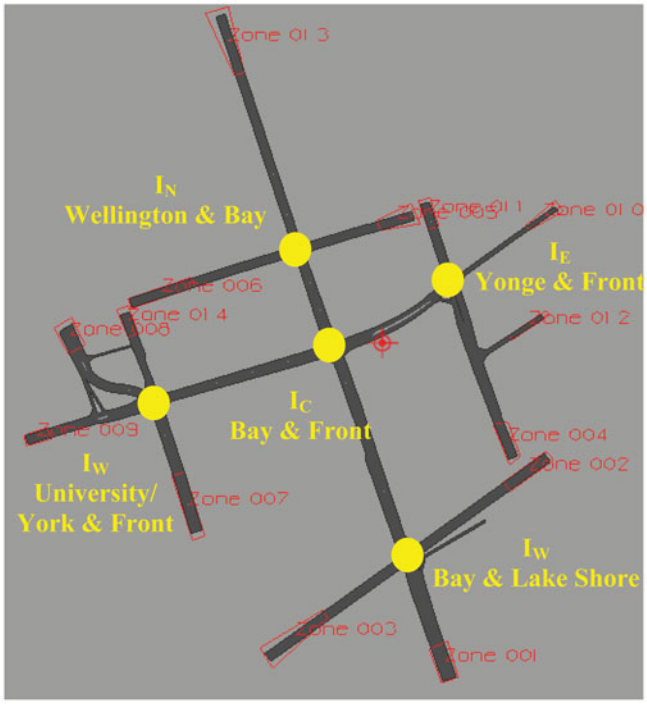


Figure 5 Medium application—testbed network.

From/To	1	2	3	4	5	6	7	8	9	10	11	12	13	14	Total
1	0	0	94	48	0	0	0	0	0	0	0	0	431	0	573
2	50	0	1718	1	0	0	0	1	2	1	1	1	67	69	1911
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	204	0	2	0	0	0	0	0	28	58	1	1	1	1	296
5	0	0	0	1	0	622	0	0	2	19	7	1	19	48	719
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	599	0	0	0	0	39	1	639
8	0	0	0	1	0	0	1309	0	1	12	49	1	19	18	1410
9	0	0	0	1	0	0	89	1	0	465	14	143	2	10	725
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	7	0	1	2	0	0	0	0	181	50	0	436	25	0	702
12	0	0	0	1	0	0	0	0	1	56	554	0	1	0	613
13	30	0	567	1	0	56	0	0	1	82	1	1	0	0	739
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Total	291	0	2382	56	0	678	1398	601	216	743	627	584	604	147	8327

Figure 6 Medium application—origin–destination demand matrix.

EXPERIMENTAL SETUP AND RESULTS 2: TWO-DIMENSIONAL PROTOTYPE NETWORK

This section demonstrates the applicability and feasibility of the RL-based ATSC presented in a prototype implementation on a network of five signalized intersections in downtown Toronto. The findings of the preceding section furnished important information for designing the parameters of RL-based ATSC in this section. The learning method, state definition, action definition, reward definition, and action selection method recommended in the preceding section are adopted in this section.

Testbed Network

RL-based ATSC is tested on a simulated network of five intersections in the heart of the financial district of downtown Toronto. Paramics is used to build the testbed network as shown in Figure 5. In this implementation, the learning process is terminated after 100 simulation runs (each is 1 hour). The origin–destination (OD) demand is estimated from the turning movement counts for the afternoon peak hour collected from the City of Toronto in 2009. The output of the OD estimation process is resulted in almost 8,300 vehicles fed into the simulation

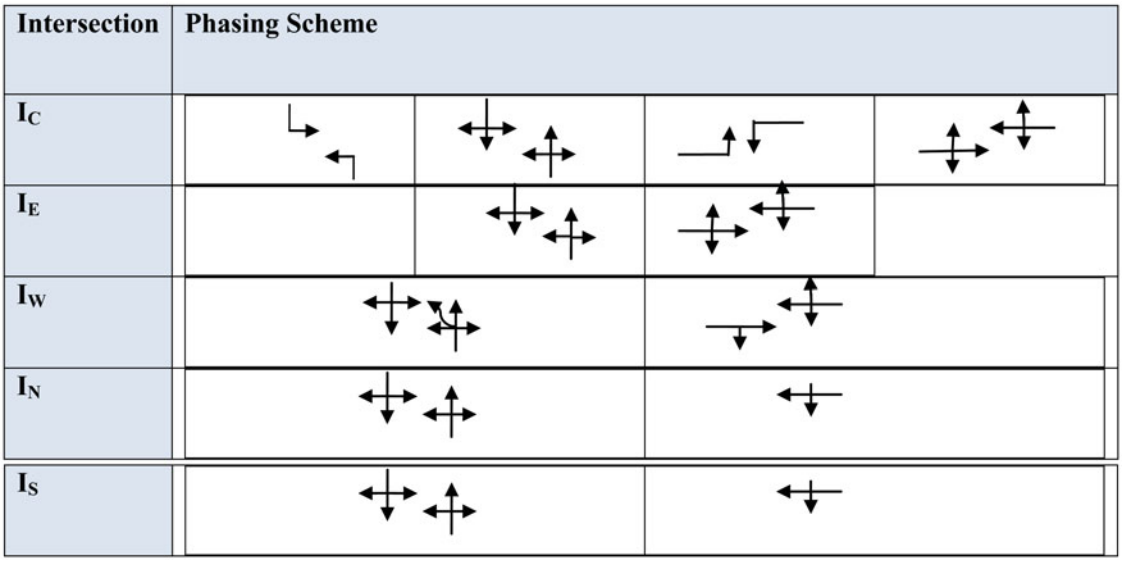


Figure 7 Medium application—Phasing scheme design.

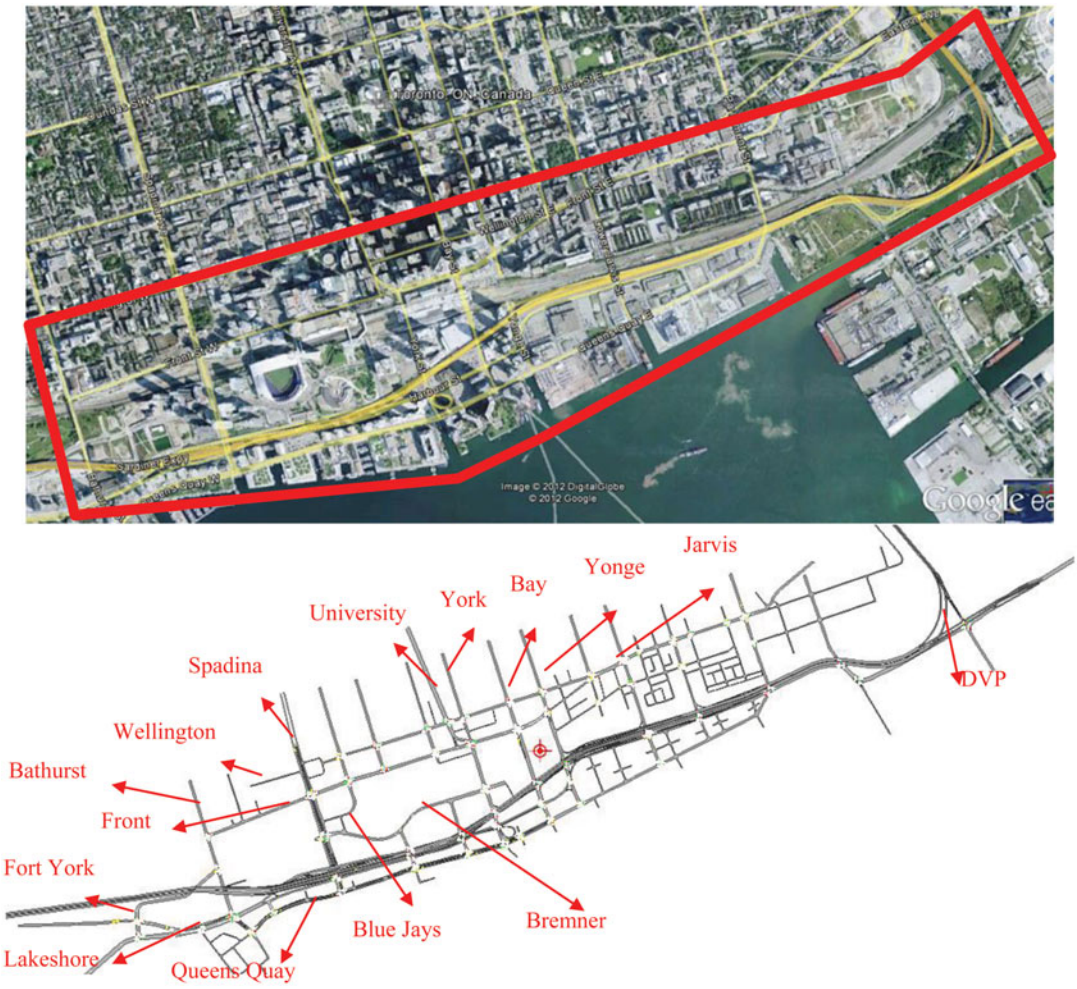


Figure 8 Large application—testbed network.

model, as shown in Figure 6. Two demand levels are modeled in this article; one represents the actual observed demand, and the other represents a 50% increase in the demand level. The increased demand scenario reflects near future conditions, given the constantly increasing population and employment and the Toronto condominium boom[2] in downtown Toronto; or it can reflect special events (e.g., sports, parades, etc.). This scenario can also reflect construction and work zones as it is currently under consideration in the Front Street reconfiguration from Bay Street to York Street (City of Toronto, 2012). All field data available are hourly data. The phasing scheme for each intersection is designed as shown in Figure 7.

Benchmarks

We compare to more traditional fixed time and actuated control methods. The fixed time signal plan is optimized using Webster (1958) method. NEMA-actuated control is applied for all the intersections in the network. Progression is maintained in the tested network by employing the City of Toronto offset values at each intersection. Each demand level may result in different optimized fixed timing signal plan. A protected left-turn phase in the SB and NB direction for intersection Ic shown in Figure 7 is considered only in the high-demand scenario because the actual demand case does not warrant a protected left-turn phase.

Results and Discussion

Table 5 presents the average delay per vehicle and LOS for each intersection using RL, fixed-time, and actuated control systems. It is found that RL-based ATSC consistently outperforms the fixed-time and actuated controls for all intersections. The effect of RL-based ATSC is more evident for the intersections at the boundaries, for closely spaced intersections, and in the high-demand case. This is because of the ability of RL-based ATSC to "meter" traffic by optimizing both the green time for each phase and the phasing sequence for each intersection that minimizes the frequency of cross-blocking and hence minimizes the

[2]Dynamics of the Toronto Condominium Boom, accessed April 24, 2012: http://ellidavis.com/toronto-real-estate-news/2012/01/toronto-condominium-boom

Table 5 Medium application—Intersection average delay.

Demand level	Control system	Average delay				
		IC	IE	IW	IN	IS
Actual	Fixed-time	34.40	19.39	13.50	21.62	13.62
	Actuated	31.99	18.36	12.80	22.76	13.59
	RL	27.06	17.43	12.24	17.51	12.80
	Percent improvement vs. fixed time	21.3%	10.1%	9.3%	19.0%	6.0%
	Percent improvement vs. actuated	15.4%	5.1%	4.3%	23.1%	5.8%
High	Fixed-time	92.38	90.81	34.31	45.79	49.49
	Actuated	89.57	86.93	30.58	41.93	45.40
	RL	75.69	64.89	30.43	32.36	28.03
	Percent improvement vs. fixed time	18.1%	28.5%	11.3%	29.3%	43.4%
	Percent improvement vs. actuated	15.5%	25.4%	0.5%	22.8%	38.3%

average delay. This feature is important, particularly in high-demand cases that could cause spill-back.

EXPERIMENTAL SETUP AND RESULTS 2: LARGE-SCALE NETWORK

To ensure scalability of the system and extendibility of the control logic, the RL-based ATSC is tested on a simulated network of the lower downtown Toronto network (see Figure 7). The lower downtown of Toronto is the core of the City of Toronto. The lower downtown of Toronto in this study is bounded to the south by the Queens Quay corridor, to the west by the Bathurst Street to the East by the Don Valley Parkway (DVP), and to the North by Front Street. Toronto is the oldest, densest, most diverse area in the region, and its downtown core contains one of the highest concentrations of economic activity in the country. This article demonstrates large-scale application of RL-based ATSC on a simulated replica of the lower downtown core. A base-case simulation model for the lower downtown core was originally developed using Paramics, a microscopic traffic simulator, in the ITS Centre and Testbed at the University of Toronto for the year 2006. In this application, the model is further refined to reflect the signal timing sheets provided by the City of Toronto.[3] The analysis period considered in this application is the morning peak hour, which has around 25,000 vehicular trips.

Benchmarks

It is typically difficult to find a benchmark for large-scale traffic signal control problems, given that the operational details of most traffic control systems are not easily available due to obvious commercial reasons. The performance of RL-BASED ATSC approach is compared to the Base Case (BC) scenario in which traffic signals, as defined and operated by the City of Toronto, are a mix of fixed-time control, semi-actuated control, and SCOOT control as shown in Figure 9. It is worth noting that due to the limited technical details about the operation of SCOOT, it is approximated in this thesis as an enhanced, fully actuated control in which loop detectors are placed on all approaches and the extension times are conducted second-by-second.

[3]Contact person is Rajnath Bissessar, City of Toronto–Transportation Services, Manager of the UTCS.

Results and Discussion

The results are reported for BC control systems (existing conditions) and RL-based ATSC. The performance of each control system is evaluated based on the following measures of effectiveness:

- Average delay per vehicle (sec/veh).
- Average maximum queue length per intersection (veh).
- Average standard deviation of queue lengths across approaches (veh).
- Number of completed trips.
- Average CO_2 emissions factors (g/km).
- Average travel time for selected routes (min).

Table 6 compares the performance of the BC against the RL-based ATSC system.

Table 6 leads to the following findings:

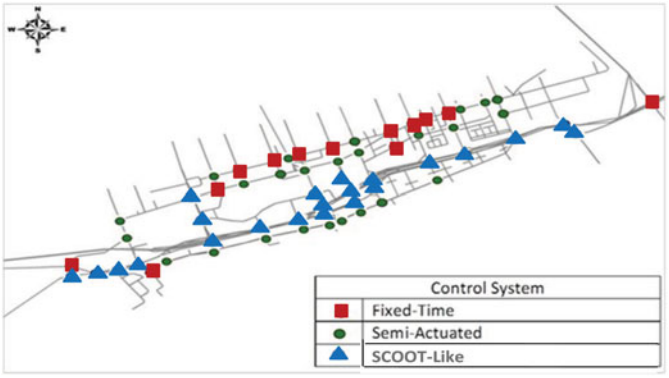


Figure 9 Large application—currently implemented signal control systems.

Table 6 Large application—Network-wide MOE.

System MOE	BC	RL-based ATSC	% Improvements RL-based ATSC Vs. BC
Average Intersection Delay (sec/veh)	35.27	25.72	27.06%
Throughput (veh)	23084	23732	2.81%
Avg Queue Length (veh)	8.66	6.60	23.77%
Std. Avg. Queue Length (veh)	2.12	1.62	23.37%
Avg. Link Delay (sec)	9.45	8.50	10.07%
Avg. Link Stop Time (sec)	2.74	2.57	5.95%
Avg. Link Travel Time (sec)	16.81	15.81	5.97%
CO2 Emission Factor (gm/km)	587.28	421.34	28.26%

- The RL-based ATSC results in lower average delay, higher throughput, shorter queue length, and stop time compared to those from the base case. The most notable improvements are the average delay (27% RL-based ATSC vs. BC), average and standard deviation of average queue length (23% RL-based ATSC vs. BC), and CO_2 emission factors (28% RL-based ATSC vs. BC).
- In fact, the tangible savings in the standard deviation in the queue length are interesting because this means balanced queue among all intersection approaches.

It is important to study the effect of various control systems on the travel time and travel time variability for selected key routes in the lower downtown core of Toronto.

Route travel times and standard deviation in travel time for the BC and RL-based ATSC scenarios for eight key routes are presented in Table 7. The analysis of Table 7 leads to the following conclusions:

Table 7 Large application—Route travel times.

System Route	BC	RL-based ATSC	% Improvements RL-based ATSC Vs. BC
1- Gardiner EB	5.14	4.98	3.18%
St Dev	1.15	0.86	0.25
2- Gardiner WB	4.42	4.27	3.35%
St Dev	0.20	0.15	26.52%
3- Front EB	10.65	9.13	14.28%
St Dev	2.15	1.22	43.26%
4- Front_WB	5.55	5.34	3.81%
St Dev	0.92	0.79	13.39%
5- LakeShore EB	16.31	13.28	18.60%
St Dev	3.74	1.37	63.38%
6- LakeShore WB	10.31	9.07	12.02%
St Dev	1.03	0.69	33.30%
7- LakeShore EB to Spadina NB	10.94	10.86	0.74%
St Dev	3.75	3.07	18.25%
8- LakeShore EB University to Ave NB	12.05	10.24	15.04%
St Dev	2.81	1.66	40.80%

- RL-based ATSC outperforms BC in almost all cases; the percentage improvements ranges from 3% in route 1 to 15% in route 5.
- From observing the temporal distribution of route travel time across the simulation hour, it is generally found that RL-based ATSC is stable and exhibits less variation compared to the BC scenario. While the BC scenario exhibits the highest variability in travel time (as shown in the standard deviation values in Table 7), RL-based ATSC shows more stable route travel times in all routes.

In terms of the computational complexity, each agent (intersection) converges to the optimal policy with different convergence speed. The average time required to converge to the minimum average delay per intersection is 60 simulation runs (1 hour each). The computational time for each learning step (1 simulation-sec) is 4.2 msec.

CONCLUSIONS AND FUTURE WORK

In this article, a generic reinforcement learning software is developed and applied to an adaptive traffic signal control of an isolated intersection. Paramics, a microsimulation simulation platform, is used to evaluate the adaptive traffic control system on a major intersection in downtown Toronto using actual observed traffic data. This article endeavors to identify the best design of the RL-based adaptive traffic signal control system for an isolated intersection. More specifically, the article investigated the following parameters/dimensions: (1) RL learning method, (2) traffic state representation, (3) action selection method, (4) traffic signal phasing scheme, (5) reward definition, and (6) variability of flow arrivals to the intersection. In addition, the effect of the best design of RL-based ATSC system is tested on a large-scale application of 59 intersections in downtown Toronto and the results are compared versus the base-case scenario of signal control systems in the field, which are mixes of pretimed and actuated controllers. In terms of the RL methods, it is found that there is no significant difference in the performance of Q-Learning and SARSA. On the other hand, TD(λ) performance is worse with higher values of (λ). Although it is found that RL generally outperforms the pretimed control regardless of the state definition, considering (1) the queue lengths at the red phases and (2) the arrivals to the green phase is found to yield the best state representations across all the experiments. Fixed phasing sequence is recommended in case of uniform arrival rate; however, in case of variable arrival rates, variable phasing sequence is recommended. The best reward function is the reduction in the cumulative delay that resulted in the minimum average delay, average queue length, and average number of stops. In terms of action selection method, synergizing

∈-greedy and softmax methods allows for faster convergence and better online performance. In general, RL-based ATSC is performing more robustly than the pretimed and actuated control in which the same average delay is obtained regardless of the arrival profiles. The effect of RL in signal control is more vivid in the variable demand profile scenario, which results in 48% saving in the average delay. This article also presented the applicability of the RL-based ATSC in a two-dimensional prototype network of five intersections in downtown Toronto using Paramics. To replicate realistic traffic conditions in the prototype implementation, two demand levels were modeled; one represented the actual observed demand, while the other represented a 50% increase in the total demand. The performance of RL-based ATSC modes were compared against the fixed-time and traffic responsive actuated control. The analysis of these experiments led to the following conclusions: (1) RL-based ATSC consistently outperformed the fixed-time and actuated control regardless the demand level in terms of the savings in average intersection delay, and (2) the effectiveness of the RL-based ATSC was found to be more vivid in the case of high demand level. The findings of the isolated intersection and the small prototype network experiments described here form important information for designing RL-parameters in the large-scale application urban network of 59 intersections. RL-based ATSC algorithms resulted in lower average delay, throughput, queue length, and stop time compared to those from the base case. Our ongoing research extends the work to developing a Multi-Agent Reinforcement Learning (MARL)-based ATSC system in which each controller (agent) coordinates its action with its surrounding neighbors (agents). In addition, the ongoing work is focused on integrating RL-based ATSC with an off-the-shelf field controller (ATC-1000 controller) that is used by many North American cities, including Burlington and Toronto, Canada, following NTCIP standards. RL-based ATSC basically override the actuated operation of the field controller by setting the controller's phase parameters (e.g., "Hold" and "Omit") with appropriate values at the appropriate time steps. However, the best detection technology to be considered is still under investigation. The authors acknowledge the importance of comparing this system against the currently implemented adaptive traffic signal control systems, such as SCOOT. This is, however, the next step to extend this research, as we are currently examining the potential of using a hardware-in-the-loop simulation approach (HILS) to possibly integrate a SCOOT box with Paramics to use it as a benchmark for our work. Also, integrating transit signal priority and pedestrians consideration are important directions for our future research.

FUNDING

The authors gratefully acknowledge the financial support of Connaught and OGS Scholarships from University of Toronto. This research was enabled by the Toronto ITS Centre.

REFERENCES

Abdulhai, B., & Kattan, L. 2003. Reinforcement learning: Introduction to theory and potential for transport applications. *Canadian Journal of Civil Engineering*, **30**, pp. 981–991.

Abdulhai, B., Pringle, R., & Karakoulas, G. J. 2003. Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering*, **129**, 278–285.

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

Balaji, P. G., German, X., & Srinivasan, D. 2010. Urban traffic signal control using reinforcement learning agents. *IET Intelligent Transport Systems*, **4**, 177–188.

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

Bingham, E. 2001. Reinforcement learning in neurofuzzy traffic signal control. *European Journal of Operational Research*, **131**, 232–241.

Camponogara, E., & Kraus, W., Jr. 2003. *Distributed learning agents in urban traffic control*. Presented at the 11th Portuguese Conference on Artificial Intelligence.

Chen, B., & Cheng, H. H. 2010. A review of the applications of agent technology in traffic and transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, **11**, 485–497.

City of Toronto. 2012. *Changes to Front Street at Union Station*. http://www.toronto.ca/involved/projects/frontunion/pdf/front-st-ea-pic2-all-comp4-sec-orient.pdf

De Oliveira, D., Bazzan, A. L. C., da Silva, B. C., Basso, E. W., Nunes, L., Rossetti, R., de Oliveira, E., da Silva, R., & Lamb, L. 2006. *Reinforcement learning-based control of traffic lights in non-stationary environments: A case study in a microscopic simulator*. Presented at EUMAS06, December 4–7, Beja, Portugal

de Queiroz, M. S., de Berrdo, R. C., & de Pádua Braga, A. 2006. Reinforcement learning of a simple control task using the spike response model. *Neurocomputing*, **70**, 14–20.

El-Tantawy, S., & Abdulhai, B. 2010. Towards multi-agent reinforcement learning for integrated network of optimal traffic controllers (MARLIN-OTC). *Transportation Letters*, **2**, 89–110.

Farges, J. L., Henry, J. J., & Tufal, J., 1983. *The PRODYN real-time traffic algorithm*. Presented at the 4th IFAC/IFIP/IFORS Symposium on Control in Transportation Systems, September 25–26, Koblenz, Germany.

Gartner, N. H. 1983. OPAC: A demand-responsive strategy for traffic signal control. *Transportation Research Record: Journal of the Transportation Research Board*, **906**, 75–81.

Gosavi, A. 2003. *Simulation-based optimization: Parametric optimization techniques and reinforcement learning*. Dordrecht, The Netherlands: Springer.

Head, K. L., Mirchandani, P. B., & Sheppard, D. 1992. Hierarchical framework for real-time traffic control. *Transportation Research Record*, **1360**, 82–88.

Hunt, P. B., Robertson, D. I., Bretherton, R. D., & Winton, R. I. 1981. *SCOOT—A traffic responsive method of coordinating signals*. Technical report. Crowthorne, England: Transport and Road Research Laboratory.

Jacob, C. 2005. *Optimal, integrated and adaptive traffic corridor control: A machine learning approach*. Department of Civil Engineering, University of Toronto, Toronto, Canada.

Jang, J. S. R., Sun, C. T., & Mizutani, E. 1997. *Neuro-fuzzy and soft computing*. Upper Saddle River, NJ: Prentice Hall.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence*, **4**, 237–285.

Leng, J., Fyfe, C., & Jain, L. C. 2009. Experimental analysis on SARSA () and Q () with different eligibility traces strategies. *Journal of Intelligent and Fuzzy Systems*, **20**, 73–82.

Lu, S., Liu, X., & Dai, S. 2008. *Incremental multistep Q-learning for adaptive traffic signal control based on delay minimization strategy*. Presented at the 7th World Congress on Intelligent Control and Automation, June 25–27, Chungking, China.

McShane, W. R., Roess, R. P., & Prassas, E. S. 1998. *Traffic engineering*. Upper Saddle River, NJ: Prentice Hall.

Park, B., & Qi, H. 2004. *Development and evaluation of a calibration and validation procedure for microscopic simulation models*. Final report. Virginia Transportation Research Council, Charlottesville, VA.

Quadstone Paramics. 2012. *Paramics Microscopic Traffic Simulation Software*. http://www.paramics-online.com

Richter, S., Aberdeen, D., & Yu, J. 2007. Natural actor-critic for road traffic optimisation. In *Advances in neural information processing systems*, vol. 19, pp 1169–1176. Cambridge, MA: MIT Press.

Salkham, A., Cunningham, R., Garg, A., & Cahill, V. 2008. *A collaborative reinforcement learning approach to urban traffic control optimization*. Presented at IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, December 9–12, Sydney, Australia.

Shoufeng, L., Ximin, L., & Shiqiang, D. 2008. *Q-Learning for adaptive traffic signal control based on delay minimization strategy*. Presented at IEEE International Conference on Networking, Sensing and Control, April 6–8, Sanya, China.

Sims, A. G., & Dobinson, K. W. 1979. *SCAT—The Sydney Co-ordinated Adaptive Traffic System–Philosophy and benefits*. Presented at International Symposium on Traffic Control Systems, August 6–9, Barkeley, CA.

Sutton. R. S., & Barto, A. G. 1998. *Introduction to reinforcement learning*. Cambridge, MA: MIT Press.

Thorpe, T. 1997. *Vehicle traffic light control using SARSA*. Master's project report, Computer Science Department, Colorado State University, Fort Collins, CO.

Wahba, M. 2008. *MILATRAS: MIcrosimulation Learning-based Approach to TRansit ASsignment*. Department of Civil Engineering, University of Toronto, Toronto, Canada.

Watkins, C., & Dayan, P. 1992. Q-learning. *Machine Learning*, **8**, 279–292.

Webster, F. V. 1958. Road Research Technical Paper No 39. *Traffic signal settings.* University of Michigan.

Wiering, M. 2000. *Multi-agent reinforcement learning for traffic light control*. Presented at the 17th International Conference on Machine Learning 2000, June 29–July 2, Stanford. CA.