

An Experimental Review of Reinforcement Learning Algorithms for Adaptive Traffic Signal Control

Patrick Mannion, Jim Duggan, and Enda Howley

Abstract Urban traffic congestion has become a serious issue, and improving the flow of traffic through cities is critical for environmental, social and economic reasons. Improvements in Adaptive Traffic Signal Control (ATSC) have a pivotal role to play in the future development of Smart Cities, and in the alleviation of traffic congestion. Here we describe an autonomic method for ATSC; namely Reinforcement Learning (RL). This chapter presents a comprehensive review of the applications of RL to the traffic control problem to date, along with a case study that showcases our developing Multi Agent Traffic Control architecture. Three different RL algorithms are presented and evaluated experimentally. We also look towards the future and discuss some important challenges that still need to be addressed in this field.

1 Introduction

Traffic congestion has become a major issue that is familiar to the majority of road users, and the environmental, social and economic consequences are well documented. Ever increasing vehicle usage rates, coupled with the lack of space and public funds available to construct new transport infrastructure, serve to further complicate the issue. Against this backdrop, it is necessary to develop intelligent and economical solutions to improve the quality of service for road users. A relatively inexpensive way to alleviate the problem is to ensure optimal use of the existing road network, e.g. by using Adaptive Traffic Signal Control (ATSC). Improvements in ATSC have a pivotal role to play in the future development of Smart

Patrick Mannion
National University of Ireland, Galway, e-mail: p.mannion3@nuigalway.ie

Jim Duggan
National University of Ireland, Galway, e-mail: jim.duggan@nuigalway.ie

Enda Howley
National University of Ireland, Galway, e-mail: enda.howley@nuigalway.ie

Cities, especially considering the current EU-wide emphasis on the theme of Smart, Green and Integrated Transport in Horizon 2020 [1]. Setting optimal traffic light timings is a complex problem, not easily solved by humans.

Autonomic systems exhibit several key properties such as adaptability, self-management and self-optimisation. Efforts are now underway to incorporate these highly desirable properties into future transportation networks, leading to the creation of Autonomic Road Transport Support Systems (ARTS). True ATSC capabilities are an important component of this vision. In recent years, applications of Machine Learning methods such as fuzzy logic, neural networks and evolutionary algorithms have become increasingly common in ATSC research.

The approach that will be examined in this chapter is Reinforcement Learning (RL), a field that has many potential applications in the Intelligent Transportation Systems (ITS) area. Reinforcement Learning for Traffic Signal Control (RL-TSC) has many benefits; RL agents can learn online to continuously improve their performance, as well as adapting readily to changes in traffic demand. RL-TSC systems may be classified as autonomic control systems since they exhibit many autonomic and intelligent characteristics. Flexible goal orientation via customised reward functions allows us to choose which system parameters we wish to optimise. RL-TSC agents are capable of learning ATSC strategies autonomously with no prior knowledge. Online learning once the system is deployed allows real time refinement of the control policy, and continuous self-management and self-optimisation.

Traffic control problems have been shown to be a very attractive testbed for emerging RL approaches [11], and present non-trivial challenges such as developing strategies for coordination and information sharing between individual agents. While we deal here only with the use of RL-TSC, there are other reviews of the applications of Machine Learning and Agent-based technologies in the broader ITS field that may be of interest to the reader [5, 12, 11, 15, 29].

The remainder of this chapter is structured as follows: the second section discusses the concept of RL, while the third section describes specific applications that we have investigated for this study. The following section details the design of our experimental set up, after which we present our experimental results. Finally, we conclude by discussing our findings, our plans for future work, and a number of challenges that still need to be addressed in the RL-TSC domain.

2 Reinforcement Learning Algorithms

The term Reinforcement Learning describes a class of algorithms that have the capability to learn through experience. An RL agent is deployed into an environment, usually without any prior knowledge of how to behave. The agent interacts with its environment, and receives a scalar reward signal r based on the outcomes of previously selected actions. This reward can be either negative or positive, and this feedback allows the agent to iteratively learn the optimal control policy. The agent must strike a balance between exploiting known good actions and exploring the

consequences of new actions in order to maximise the reward received during its lifetime. Q values represent the expected reward for each state action pair, which aid the agent in deciding which action is most desirable to select when in a certain state. The Q values are typically stored in a matrix, which represents the knowledge learned by an RL agent. For further detail on RL beyond the summary presented in this chapter, we refer the reader to [14, 42, 49].

2.1 Markov Decision Processes, Policies and Optimality

An RL problem is generally modelled as a Markov Decision Process (MDP), which is considered the de facto standard when formalising problems involving learning sequential decision making [49]. An MDP may be represented using a reward function R , set of states S , set of actions A , and a transition function T [36], i.e. a tuple $\langle S, A, T, R \rangle$. When in any state $s \in S$, selecting an action $a \in A$ will result in the environment entering a new state $s' \in S$ with probability $T(s, a, s') \in (0, 1)$, and give a reward $r = R(s, a, s')$.

A policy π determines the agent's behaviour in its environment. Policies provide a mapping from states to actions that guide the agent when choosing the most appropriate action for a given state. The goal of any MDP is to find the best policy (one which gives the highest overall reward) [49]. The optimal policy for an MDP is denoted π^* .

2.2 Model-based Reinforcement Learning

RL can be classified into two categories: model-free (e.g. Q-Learning, SARSA) and model-based (e.g. Dyna, Prioritised Sweeping). To implement model-based approaches successfully, it is necessary to know the transition function T [49], which may be difficult or even impossible to determine even in relatively simple domains. By contrast, in the model-free approach this is not a requirement. Exploration is required for model-free approaches, which sample the underlying MDP in order to gain knowledge about the unknown model. The use of a model-based approach in a highly stochastic problem domain like traffic control also adds unnecessary extra complexity when compared with a model-free approach [21]. Our analysis in this chapter will focus in the main on model-free approaches to this problem domain for the reasons outlined above.

One of the most popular RL approaches in use today is Q-Learning [45]. It is an off-policy, model-free learning algorithm that is commonly used in RL-TSC literature, e.g. [6, 7, 10, 19, 20, 21, 30, 38, 39, 52]. It has been shown that Q-learning converges to the optimum action-values with probability 1 so long as all actions are repeatedly sampled in all states and the action-values are represented discretely [44]. In Q-Learning, the Q values are updated according to the equation below:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha(r_t + \gamma Q_{\max_a}(s_{t+1}, a) - Q_t(s_t, a_t)) \quad (1)$$

SARSA [37] (meaning State-Action-Reward-State-Action) is another common model-free RL approach. This algorithm has also been proven to converge to the optimal value function so long as all state action pairs are visited infinitely often, and the learning policy becomes greedy in the limit [40]. SARSA is very useful in non-stationary environments where an optimal policy may never be reached, and also in situations where the use of function approximation is desired [49]. Use of SARSA is also quite common in RL-TSC literature, e.g. [13, 33, 43, 47]. When using SARSA, the Q values are updated as follows:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha(r_t + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)) \quad (2)$$

The learning rate $\alpha \in [0, 1]$ is an input parameter required for many RL algorithms, and determines by how much Q values are updated at each time step t . The learning rate is typically initialised as a low value (e.g. 0.05) and may be a constant, or may be decreased over time. Setting $\alpha = 0$ would halt the learning process altogether, which may be desirable once a satisfactory solution is reached. Selecting a value of α that is too large may result in instability and make convergence to a solution more difficult.

The discount factor $\gamma \in [0, 1]$ controls how the agent regards future rewards, and is typically initialised as a high value (e.g. 0.9). Setting $\gamma = 0$ would result in a myopic agent, i.e. an agent concerned only about immediate rewards, whereas a higher value of γ results in an agent that is more forward-looking.

2.3 Exploration vs. Exploitation

When using RL, balancing the exploration exploitation trade-off is crucial. Exploration is necessary for a model-free learning agent to learn the consequences of selecting different actions [49] and thus determine the potential benefit of selecting those actions in the future. Exploitation of known good actions is necessary for the agent to accumulate the maximum possible reward. As we have seen above, two of the most common model-free RL algorithms have been proven to converge to an optimum solution when all states are visited infinitely often (thus exploring all possible states is a condition for convergence to the optimum policy). While this is not practical in real world transportation problems, it is logical that the RL agent will make better informed decisions, and thus perform better, when an efficient exploration strategy is in place that allows for sufficient exploration. Conversely, selecting an inappropriate exploration strategy may result in the agent getting stuck in a local optimum and not converging to the best possible solution. Excessive exploration limits the agent's capacity to accumulate rewards over its lifetime, and may reduce performance. Two algorithms that are commonly used to manage the exploration exploitation trade-off are ϵ -greedy and softmax (Boltzmann) [49].

3 Reinforcement Learning for Traffic Signal Control

In this section, we discuss the development of RL-TSC approaches by various researchers and the challenges associated with implementation, as well as the current state of the art. The use of RL as a method of traffic control has been investigated since the mid-1990s, but the number of published research articles has increased greatly in the last decade, coinciding with a growing interest in the broader ITS field among the research community. The potential for performance improvements in Traffic Signal Control offered by RL when compared to conventional approaches is vast, and there are many published articles reporting promising results, e.g. [4, 6, 13, 21, 26, 28, 48]. Thorpe describes some of the earliest work in this area, and even at this initial stage in the development of RL-TSC it offered significant improvements over fixed signal plans [43].

3.1 Problem Formulation

The traffic control problem may be considered as a Multi Agent System (MAS), where each agent is autonomous and responsible for controlling the traffic light sequences of a single junction. In the context of RL-TSC, this scenario is described by the term Multi Agent Reinforcement Learning (MARL), which consists of multiple RL agents like those detailed in section 2, learning and acting together in the same environment. The scenario is a Partially Observable Markov Decision Process, as it is impossible for agents to know every detail about the environment in large scale transportation networks in the real world.

The MAS paradigm is inherently very suitable for the management of transportation systems [12], and also for transport simulations. Modern transportation networks may contain thousands or millions of autonomous entities, each of which must be represented in simulations. Vehicles, drivers, pedestrians and traffic signal controllers can all be modelled as agents. Use of MAS allows large scale representation of these agents, while also allowing fine-grained detail for each individual agent if desired. Additionally, MAS is innately robust and highly scalable [14]; there is no single point of failure and new agents can easily be inserted into the system. Wooldridge [51] gives a good overview of the main concepts in Multi Agent Systems (MAS), which may be of interest to the reader.

A naïve way of using RL to control traffic signals is to train a single RL agent to control all junctions in the system. However, this approach is not feasible when considering large networks, due to a lack of scalability and the huge number of potential actions that may be selected [9]. Thus, MARL is the standard approach in RL-TSC research, as it scales much better than a single agent controlling an entire network.

3.2 *Challenges of Applying RL to Traffic Signal Control*

A significant challenge in RL-TSC is implementing co-ordination and information sharing between agents [11, 21, 28]. The effect of any single agent's actions on the environment is also influenced by the actions of other agents [14]. Single agents are self-interested and will seek only to maximise their personal rewards. For example, a control policy selected by an agent may result in a local optimum in terms of traffic movements, but may have a detrimental effect on traffic flows in the network as a whole, limiting the effectiveness of the other agents. Thus, having multiple agents that are primarily greedy or self-interested is not necessarily the best option, and some sort of co-ordination or information sharing mechanism is necessary to implement systems relevant to the real world.

One of the main difficulties associated with determining optimal traffic control policies is the sheer scale of the problem. With respect to RL, the term Curse of Dimensionality is frequently used to describe the difficulty of dealing with the plethora of information that must be handled [21, 34, 42, 49]. As each individual intersection has its own controller agent, the problem complexity increases vastly in larger road networks. When implementing coordination between agents, this increase in dimensionality is even more pronounced. In many commonly used MARL coordination methods, each agent has to keep a set of tables whose size is exponential in the number of agents [21]. Later, we will examine some of the approaches used by researchers to tackle these formidable problems. Model-free RL methods become even more attractive as network size increases, as the absence of a model reduces the computational complexity when compared with model-based methods. The continual increase in available computational power is another factor that will make this challenge easier to deal with in the future.

3.3 *State Definitions*

One of the most important issues to be considered when designing an RL-TSC system is that of determining how the state of the environment will be represented to the agent. Various different approaches to defining the environment state in RL-TSC problems have been proposed in the literature, with each having its own relative merits. Queue length is one of the most common state definitions used in RL-TSC [6], and has been used as a means of state representation in many RL-TSC research papers, e.g. [3, 4, 6, 16, 19, 20, 21]. Wiering [48] proposed a vehicle-based state definition, based on the expected total waiting time for a vehicle to reach its destination. Delay-based approaches are also considered in e.g. [7, 9, 23, 28, 41, 50, 8]. In [10] the traffic state is estimated by considering both queue length and traffic flow rate, while [13] presents an approach based on queue length and time elapsed since the previous signal change.

3.4 *Reward Function Definitions*

Choosing an appropriate reward function is equally important as the choice of state representation when designing an RL-TSC agent. These issues are somewhat inter-related, in that the reward received by an agent is often related to the difference in utility between the current and previous states, i.e. the agent is rewarded for actions that improve the state of the environment. Equivalently, we can think of optimising some parameter as the agent's objective, and this objective is specified in the reward function.

Many different objectives have been considered by authors when defining the reward function used by RL-TSC agents. These include Average Trip Waiting Time (ATWT) / trip delay, Average Trip Time (ATT), Average Junction Waiting Time (AJWT), Junction Throughput/Flow Rate (FR), achieving Green Waves (GW), Accidents Avoidance (AA), speed restriction, fuel conservation and Average Number of Trip Stops (ANTS). Brys et al [13] have proposed delay squared as a possible alternative reward signal, the idea being that large delays will be punished more severely than small delays. Their results suggest that the use of delay squared as a reward signal may result in faster learning rates than simple delay.

It should be noted that many of the proposed reward definitions above rely on information that is easy to obtain in a simulation, but quite difficult or impractical to obtain in a real world setting using current technologies. Data about ATWT, ATT, ANTS, fuel usage or emissions could not be collected reliably without some form of Vehicle to Junction (V2J) communication, whereby a vehicle could report these parameters to a controller agent via wireless communications. This research area is still under development, and is not yet mature enough for widespread deployment. Also, all or nearly all vehicles using the network would have to have V2J capabilities for the statistics to be useful in assigning the agent a reward.

Dresner and Stone [18] have suggested using RL in their Autonomous Intersection Management architecture with V2J communications. This approach treats the intersection as a marketplace where vehicles pay for passage or pay a premium for priority, and the controller agent's goal is to maximise the revenue collected. In future, revenue collected could be used in reward functions for RL-TSC.

3.5 *Performance Metrics*

Evaluating the merits of any traffic control system requires suitable performance metrics. In RL-TSC these parameters are doubly important, as an agent must receive an assessment of its own performance in order to learn. In some cases, these can be the same or similar parameters used in the environment state definitions or the reward functions of the individual agents, but aggregated on a network-wide level to give an assessment of an RL-TSC system. Some performance metrics used in transportation and RL-TSC literature include reduction in emissions or fuel consumption, number of stops in a trip, percentage of stopped vehicles, delay time/ATWT, vehicle

density in various parts of the network, junction queue lengths, mean vehicle speed and junction throughput.

3.6 Single Agent Reinforcement Learning Approaches

The most basic applications of RL to Traffic Control in the literature consider only a single junction controlled by a single agent, in what is called Single Agent Reinforcement Learning (SARL). While this approach may be appropriate for signalised intersections that are isolated and not part of a larger control network (e.g. junctions in small towns), it is not suitable for deployment in large urban networks. As SARL approaches do not exploit the full potential of RL-TSC they will not be discussed in great detail, although they still prove its potential to outperform conventional approaches. There are a number of published works describing RL controlled isolated signalised junctions [6, 16, 19, 30, 47].

One such approach is that of Abdulhai et al [6], where the authors present a case study of the application of Q-Learning to control an isolated two-phase signalised junction. The Q-Learning agent was found to perform on a par with pre-timed signals on constant or constant-ratio flows, and to significantly outperform pre-timed signals under more variable flows. The authors attributed the latter result to the ability of the agent to adapt to changing traffic conditions.

In [19], the authors present a study based on a single junction in the downtown Toronto area. The designs of three Q-Learning agents with different state representations are outlined, and the performance of these agents is compared to a reference Webster-based [46] fixed time signal plan. The state definitions considered were based on: (1) Arrival of vehicles to the current green direction and queue length at red directions, (2) queue length and (3) Vehicle Cumulative Delay. The authors reported that Q-Learning outperformed the reference fixed signal plan, regardless of the state representation used or traffic conditions. The Q-Learning agent using Vehicle Cumulative Delay for state representation was found to produce the best results in heavy traffic conditions.

3.7 Multi Agent Reinforcement Learning Approaches

Most authors have focused on developing approaches based on MARL, i.e. networks consisting of multiple signalised intersections, each controlled by an independent RL agent. These cases are much more relevant to real world traffic signal control problems than the SARL cases described above. One of the most significant early works in Multi Agent RL-TSC is that of Wiering [48]. Several authors have since extended Wiering's approach or used his algorithms as a benchmark for their own approaches, e.g. [8, 9, 23, 28, 41, 50]. Wiering developed a model-based approach to RL-TSC, and presents three such algorithms: TC-1, TC-2 and TC-3. He found that

the RL systems clearly outperformed fixed time controllers at high levels of network saturation, when testing on a simple 3 x 2 grid network. Another interesting aspect of this research is that a type of co-learning is implemented; value functions are learned by signal controllers and driver agents, and the drivers also learn to compute policies that allow them to select optimal routes through the network.

Steingröver et al [41] extend the work of Wiering, by introducing a basic form of information sharing between agents. Three new traffic controllers are described by the authors: TC-SBC (Traffic Controller - State Bit for Congestion), TC-GAC (Traffic Controller - Gain Adapted by Congestion) and TC-SBC+GAC which combines the latter approaches. In TC-SBC, a congestion bit is added to the state tuple which represents the amount of traffic congestion at neighbouring intersections. One major disadvantage of TC-SBC is that it increases the size of the state space, which makes the problem more difficult to compute. TC-GAC uses congestion information from neighbouring junctions when estimating the optimal action selection. An advantage of this method is that it does not increase the size of the state space, but it never learns anything permanent about congestion in the network. The algorithms are tested on a simple grid network, under both fixed and variable flows. All three methods proposed were found to outperform Wiering's TC-1, and TC-GAC was found to provide the best performance under variable traffic flows. A further extension of this work is presented by Isa et al in [23]. Two new algorithms called TC-SBA (Traffic Controller - State Bit for Accidents) and TC-SBAC (Traffic Controller - State Bit for Accidents and Congestion) are presented, both based on TC-1. In TC-SBA, similar to TC-SBC, an extra bit is added to the state tuple, this time representing whether the lane ahead is obstructed by an accident. TC-SBAC adds both the state and accident bits to TC-1. Both of these approaches suffer from the same problem as TC-SBC, namely a substantial increase in the state space. In the case of TC-SBAC, the state space is four times larger than that of the original TC-1 algorithm.

In [3], Abdoos et al present a Multi Agent RL-TSC implementation, tested on a relatively large abstract network consisting of 50 junctions. The authors based their algorithm on Q-learning, which was tested against reference fixed time signal plans. At each junction, average queue length over all approaching links is used for state representation, and action selection is by means of selecting the green time ratio between different links. The proposed algorithm is found to offer a substantial performance improvement over fixed time control, greatly reducing delay times in the network.

Salkham and Cahill [38] developed a Multi Agent RL-TSC system, Soilse. This approach utilises a pattern change detection (PCD) mechanism that causes an agent to relearn based on the degree of change detected in traffic flows. A collaborative version, SoilseC is also described, which adds a collaborative reward model to the Soilse architecture to incorporate reward information exchanged between agents. The authors test their system on a simulated network based on Dublin city centre consisting of 62 signalised junctions, using assumed traffic flow data. Two baseline fixed time control schemes were also tested. Soilse and SoilseC were shown to outperform both baselines in terms of vehicle AWT and average number of stops.

This work is significant in that a large-scale test of RL-TSC on a real urban traffic network is presented.

3.8 Coordinated MARL Approaches

A natural extension to the approaches described above is to use MARL to achieve decentralised and coordinated traffic signal control. Use of RL and game theoretic approaches to achieve coordination in traffic control agents is certainly plausible [11], and this is an active research theme in RL-TSC.

Kuyer et al [28] extend the work of Wiering [48] to develop a coordinated model-based RL-TSC system using the Max-Plus algorithm as a coordination strategy. Max-Plus is used to compute the optimal joint action by means of message passing between connected agents. However, this approach is very computationally demanding, as agents must negotiate when coordinating their actions. The authors report improved performance when comparing their approach to that of Wiering [48] and Bakker et al. [8].

El-Tantawy and Abdulhai [20], and El-Tantawy et al [21] present a coordinated Multi Agent RL-TSC architecture called MARLIN-ATSC. This is a model-free architecture based on Q-Learning, where the state definition is based on queue length, and the reward definition is based on Total Cumulative Delay. The authors deal with the dimensionality problem by utilising the principle of locality of interaction among agents [31], and the modular Q-Learning technique [32]. The former principle means that each agent communicates only with its immediate neighbours, while the latter allows partitioning of the state space into partial state spaces consisting of only two agents. This approach significantly reduces the complexity of the problem, while still producing promising results.

In [20], the architecture is tested on a five intersection network against a Webster-based fixed time plan, as well as uncoordinated Q-Learning controllers. MARLIN-ATSC was reported to outperform both the fixed timing and uncoordinated RL approaches, resulting in a reduction in average delay. The proposed architecture was also found to offer improved convergence times when compared to a network of independent learning agents.

The authors extend their approach in [21], where the system is tested on a simulated network of 59 intersections in Downtown Toronto, using input data provided by the City of Toronto, including traffic counts and existing signal timings. In this instance, MARLIN-ATSC outperformed the currently implemented real world control scheme as well as the uncoordinated RL approach, resulting in a reduction in average delay, average stop time, average travel time, maximum queue lengths and emissions. The results presented in [21] are very encouraging, and this work is one of the largest and most realistic simulation tests of an RL-TSC approach to date, due to the use of a real urban network, along with real world traffic data and signal timings.

3.9 Multi Objectivity

Multi-Objectivity is an emerging research theme in RL-TSC that has received some attention in recent years [13, 22, 25, 24, 26]. In RL-TSC, this approach seeks to optimise a number of parameters in the network at once, the idea being that considering multiple parameters may lead to better solutions and convergence times, as well as allowing consistent performance even under varying demand levels. This is typically achieved by expanding the agent's reward function definition to include multiple parameters.

The majority of traditional traffic control methods are single-objective [25], seeking optimal solutions based on a single parameter only. As we have seen in section 3.5 above, some reward function definitions are more suited to certain traffic patterns, so combining several definitions may be beneficial where the demand is highly variable. Multi-Objective RL is comparable to Single-Objective RL in terms of time complexity [26]; this is because the only extra operation required to extend a Single-Objective approach is scalar addition to determine the combined reward signal. This means that the potential performance improvements can be gained quite cheaply in terms of computational complexity.

Houli et al [22] present a Multi-Objective RL-TSC algorithm, which is tuned for three different scenarios: light, medium and heavy traffic. Each level of traffic has a corresponding Q function. However, adaption occurs offline, as one Q function is activated at a time based on the number of vehicles entering the network in a given timeframe.

Brys et al [13] observed in their experiments that the objectives throughput and delay are correlated. They implemented a Multi-Objective RL-TSC algorithm by replacing the single objective reward signal with a scalarised signal, which was a weighted sum of the reward due to both objectives. The onus is on the user, however, to select weightings that will give good results. They report that the proposed multi-objective approach exhibits a reduced convergence time, as well as decreasing the average delay in the network when compared to a single objective approach. While performance gains can be seen from this Multi-Objective approach, selecting appropriate values for the weightings of rewards was reported to be a very time consuming task.

A more advanced approach is presented by Khamis and Gomaa [24, 26], and Khamis et al [24]. In [25, 24] the authors develop Multi-Objective RL-TSC systems with weighted sum reward functions. Multiple reward criteria are considered in the combined reward function (FR, ATWT, ATT, AJWT, safety/speed control). The systems were tested on simple grid-type networks with varying demand levels. The authors report that their approach outperforms Wiering's TC-1 [48] algorithm, resulting in a reduction in the average number of stops and waiting time in each trip [24], as well as increasing the average speed for each trip [25]. In [26] the authors extend their work further, considering a total of seven different parameters in the reward function proposed. This extended approach is tested against TC-1, SOTL [17] (Self-Organising Traffic Lights) and a Genetic Algorithm proposed by Wiering [50]. The proposed Multi-Objective approach outperforms the other three control

schemes, again showing a significant reduction in ATWT and number of stops, as well as an increase in the average speed of vehicles in the network.

3.10 Function Approximation

In complex environments, it is not possible for the RL agent to visit each state action pair infinitely often to ensure convergence. RL literature describes a technique called Function Approximation (FA), whereby explicit tabular representations of rewards due to each state action pair are not required; instead, it is possible to generalise across different states and/or actions. By approximating state action pairs, the agent can learn reasonable behaviour more quickly than a tabular implementation [42]. In general, the values of the approximate function are defined using a set of tuneable parameters [49].

Prashanth and Bhatnagar [34, 35] present the first published application of FA to RL-TSC. Two RL algorithms based on Q-Learning are presented; one with full state representation (FSR), and one using FA. The latter algorithm uses features in place of FSR, namely, traffic demand and wait. Demand level is classified based on queue lengths (low, medium or high) and wait is classified based on time that a red signal has been displayed to the lane. This approach does not require precise queue length information, and it does not require a precise value of time elapsed, as time is defined as being above or below a threshold. The goal is to minimise queue lengths while ensuring fairness so that no lane has an overly long red time. In [34], the algorithms are benchmarked against reference fixed timings and SOTL [17] in four test networks. The proposed Q-Learning with FA approach consistently outperformed the other approaches tested, while also being more efficient in terms of memory usage and computation time compared to RL with FSR.

Abdoos et al [4] also applied FA to RL-TSC, using tile coding. In tile coding, the receptive fields of the features are grouped into partitions called tilings in which each element is called a tile, and the overall number of features that are present at one time is strictly controlled and independent of the input state [42]. Each intersection is controlled by a Q-Learning agent, and these agents are grouped together to be controlled by superior agents. These superior agents use tile coding as a method of linear function approximation because of the large state space involved. This hierarchical control algorithm was tested on a 3 x 3 junction grid, and was found to outperform a standard Q-Learning approach, resulting in reduced delay times in the network.

Pham et al [33] present an RL-TSC system based on SARSA that also uses tile coding as a method of function approximation. In contrast to the approach above, this system does not include any form of coordination between agents. Instead, each SARSA agent is completely independent, and tile coding is used only as a method of approximating the value function for the agent's local states.

Arel et al [7] propose a Q-Learning algorithm with function approximation for traffic signal control. The authors implement a feedforward neural network trained

using back-propagation to provide an approximation to the state-action value function. Testing was conducted on a five intersection grid, using the proposed algorithm, along with a longest queue first approach. The authors report that their Q-Learning with function approximation algorithm outperformed the longest queue first approach at high demand levels.

4 Experimental Design

We now present a case study, which is based on the RL-TSC testbed that we are currently developing. The microscopic traffic simulation package SUMO (Simulation of Urban MObility) [27] is the basis for our experimental setup. Agent logic is defined in our external MARL framework, which is implemented in Java. Each RL agent is responsible for controlling the light sequences of a single junction. The TraaS library [2] is used to feed simulation data to the agents, and also to send commands from the agents back to SUMO.

For each agent, state is defined as a vector of dimension $2 + P$, where the first 2 components are the index of the current phase and the elapsed time in the current phase. The remaining P components are the maximum queue lengths for each phase. This is a similar state definition to that of [20]. We used a mixed radix conversion to represent this state vector as a single number, which is used when setting and retrieving values in the Q values matrix. For all junctions in our simulations, the number of phases is two. We limit the maximum number of queueing vehicles considered by an agent to 20, and the maximum phase elapsed time considered is limited to 30 seconds. By imposing these limits, we reduce the possible number of states considered by an agent. Even so, there are over 27,000 possible states that arise from the ranges used.

The actions available to the agents at each time step are: to keep the currently displayed green and red signals, or to set a green light for a different phase. Phases are subject to a minimum length of 10 seconds, to eliminate unreasonably low phase lengths from consideration. There is no fixed cycle length, and agents are free to extend the current phase or switch to the next phase as they see fit. When changing phases, an amber signal is displayed for 3 seconds, followed by an all red period of 2 seconds, followed by a green signal to the next phase. Actions are selected using the ϵ -greedy algorithm, which chooses either a random action, or the action with the best expected reward, where ϵ is the probability of choosing a random action. All agents begin with ϵ set to 0.9 at the beginning of the experiments, and initialising ϵ with a high value encourages early exploration of different states and actions, as well as improving convergence times. The value of ϵ is decreased linearly over time, to a final value of 0.1 after 10 hours of simulated time. This final value promotes exploitation of the knowledge the agent has gained, but still allows for some exploration.

We tested our agents on a 3 x 3 grid network, with a total of 9 signalised junctions. We have chosen a grid type network for a number of reasons: this is the most

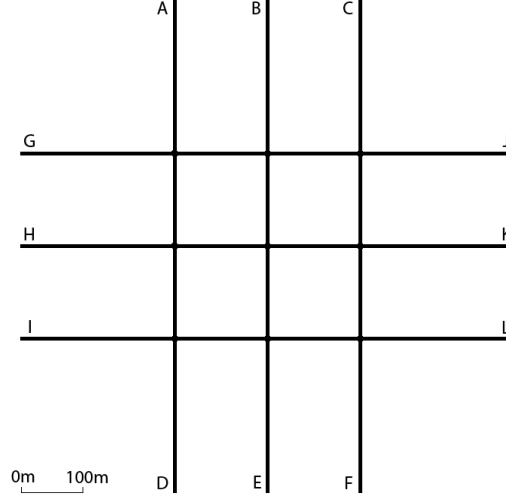


Fig. 1 The 9 Junction Test Network

common type of network used in the RL-TSC literature (e.g. [4, 7, 9, 26, 33, 34, 35, 41, 50]), which makes it easy for researchers to replicate experimental conditions and compare results. Also, many cities are based on a grid type layout (e.g. New York, Washington DC, Miami), and therefore the results presented are relevant to real urban traffic networks, and give an indication of the potential performance improvements. All lanes in our network have a maximum speed of 50 km/h. Two distinct traffic demand definitions are used during each 72 hour experiment run. The first 36 hours are used as a training period for the agents, which allows them to gain the required experience. Simple horizontal and vertical routes are defined for this period, with 200 vehicles per hour travelling on each of the routes AD, BE and CF, and 300 vehicles per hour travelling on each of the routes GJ, HK and IL. These routes are shown in Fig. 1.

In the second 36 hour period a random traffic pattern is used to simulate a more unpredictable user demand. A set of random trips were defined and processed using the SUMO DUAROUTER tool, with a frequency of 1800 vehicles per hour, and a minimum trip distance of 450m. The results presented for each RL algorithm are derived by taking an average of 10 simulation runs, to give results that are representative of the algorithm's performance.

We evaluate three different RL-TSC algorithms: RL-TSC-1, RL-TSC-2 and RL-TSC-3, all of which are based on Q-Learning. In RL-TSC-1, we define the reward as the difference between the previous and current average queue length (AQL) at the junction ($R(s, a, s') = AQL_{s'} - AQL_s$), and we have defined a queueing vehicle as one travelling below 10 km/h. The reward function for RL-TSC-2 is based on waiting times at the junction. The average waiting times (AWT) for each junction approach

are added, and the reward the agent receives is the difference between the sums of the current and previous waiting times ($R(s, a, s') = \Sigma AWT_{s'} - \Sigma AWT_s$). RL-TSC-3 has a Multi Objective reward function, which is the unweighted sum of the reward functions used in RL-TSC-1 and RL-TSC-2 ($R(s, a, s') = AQL_{s'} - AQL_s + \Sigma AWT_{s'} - \Sigma AWT_s$). The learning rate α for all agents is set to 0.08, while the value used for the discount factor γ is 0.8. A fourth agent type which implements fixed signal timing is also tested as a reference. The Fixed Timing agent has a cycle length of 60 seconds, and divides the available green time in the ratio of the fixed flows defined in the first simulation period (40% green time to the North-South phase and 60% green time to East-West phase). The Fixed Timing agent maintains the same control scheme throughout the entire 72 hour test period.

5 Experimental Results & Discussion

Results are reported for each of the four algorithms described in the previous section. The performance metrics we have chosen for evaluation of our simulation results are junction queue length, waiting times, and vehicle speeds. The values reported are averages of the results in the entire test network. In Table 1 below, a summary of the results for each control method under random flows is shown. Figures 2 and 3 show queue lengths and waiting times respectively, which are plotted for the full 72 hour test period. During the first 36 hours of each experiment, we see a gradual improvement in both waiting times and queue lengths for each of the three RL-TSC approaches, as the agents learn more about the effects of their actions on the traffic conditions. Of the three RL agent types, RL-TSC-2 offers the best performance under steady flows, with slightly lower waiting times than the Fixed Time agent by the end of the 36 hour learning period.

These graphs show a marked difference between the Fixed Time and RL-TSC control schemes under random traffic flows during the second 36 hour phase of the experiments. We see that the Fixed Time agents clearly cannot cope with the random traffic patterns in the network; this is expected as these agents are optimised for fixed ratio flows. However, all three RL-TSC based algorithms can adapt to this random traffic load, albeit with fluctuations in the queues and waiting times due to the unpredictability of the demand. From the summary of experimental results presented in Table 1, it is clear that RL-TSC-1 performs the best under these unpredictable traffic conditions, with the highest vehicle speeds, and the lowest queue lengths and waiting times. RL-TSC-2 performs slightly worse here, with lower vehicle speeds, and increased waiting times and queues compared to RL-TSC-1. RL-TSC-3 performs somewhere between the latter two algorithms; this is reasonable as it is effectively a hybrid of these approaches.

By taking an average of ten simulation runs for each agent type, we have demonstrated that RL based approaches offer consistently better performance under variable traffic flows than fixed time traffic signal plans. Each RL algorithm is also competitive with fixed traffic signals under more predictable flows, with RL-TSC-

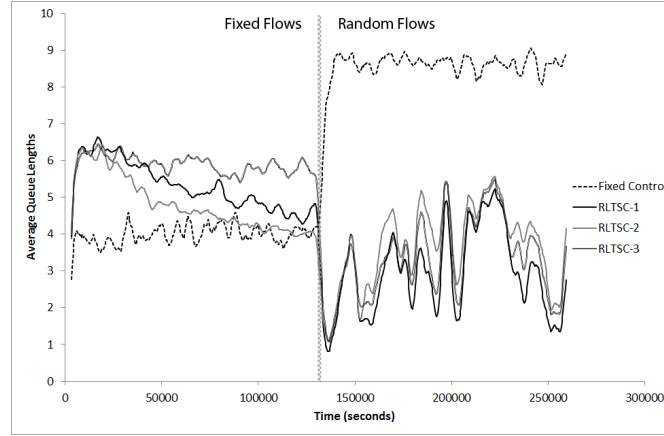


Fig. 2 Average Junction Queue Length

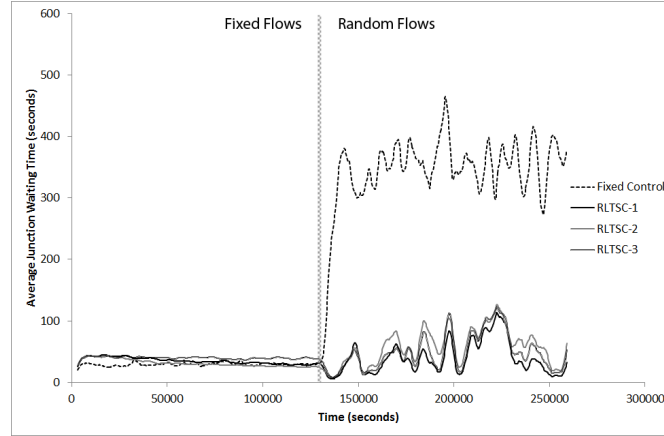


Fig. 3 Average Junction Waiting Times

2 offering the best performance under this type of flow regime. Further work is required on RLTS-1 and RLTS-3 in order for them to achieve the same performance under predictable flows as a fixed time controller. In general, our findings concur with those of Abdulhai et al in [6], who showed that a Q-Learning agent can perform on a par with pretimed signals under fixed flows, and outperform pretimed signals under more variable flows. Further work is also required on our Multi Objective algorithm RLTS-3, in order to improve its performance. These refinements may include considering different reward definitions, along with determining the optimal weighting for each of the parameters considered. Determining the optimal parameter weightings in Multi Objective reward functions is a time consuming en-

Table 1 Summary of Performance Under Random Traffic Flows (Average Values)

Algorithm	Queue Length	Vehicle Speed (km/h)	Waiting Time (seconds)
Fixed Timing	8.55	7.98	346.49
RLTSC-1	2.93	12.91	39.26
RLTSC-2	3.68	12.29	57.42
RLTSC-3	3.37	12.58	48.32

deavour, so we may consider implementing a learning based approach to determine these weightings automatically.

In summary, we recommend the use of RLTSC-2 for conditions where there are fixed flows or fixed ratio flows. RL-TSC-1 is recommended for highly variable traffic conditions, while the multi-objective RLTSC-3 algorithm performs somewhere between the two and offers good all-round performance.

6 Conclusions and Future Work

As we have outlined in this chapter, the field of RL-TSC is a very promising one. We have provided the reader with a general introduction to the concept of RL, as well as discussing the factors that need to be taken into consideration when designing MARL architectures capable of controlling traffic signals, and how various authors have addressed these challenges in their own research. In our experiments we have demonstrated the suitability of RL-TSC to deal with time varying flows and random traffic patterns when compared to a fixed time control scheme. RL-TSC has matured significantly in the last decade, and while much has been accomplished, it has yet to advance beyond theory and simulations. There have been no reported field deployments at the time of writing, but this is the ultimate goal of RL-TSC research.

Real world deployment would of course necessitate a blend between both offline and online learning to be successful. Ideally, agents would first be trained offline in a simulator, using real world network geometry and traffic demand data. Incorporating some online learning would allow the agents to adapt to changing traffic patterns, if this behaviour is desired. Finding the right balance between online and offline learning will be important when deploying RL-TSC systems in the field, and remains unaddressed in the literature. Further studies using real road networks and corresponding traffic demand data should be conducted to further this aim, preferably using the existing signal plans as a benchmark.

Questions concerning the robustness and reliability of controllers, and the effect of a failure on traffic dynamics also remain unanswered at the time of writing. One must also consider how RL-TSC will interface with other promising approaches in ITS in the future, e.g. Autonomous Intersection Management [18]. Many approaches in the literature use parameters for state and reward definitions that are not easily obtained in real traffic networks, e.g. ATWT. To apply these approaches, we

will need to develop and include the required data gathering capabilities in our traffic networks, e.g. vehicle detecting cameras, Vehicle to Junction communications, or even floating car data.

Some of the recently emerging themes in RL-TSC, e.g. Function Approximation, Multi-Objectivity and Co-ordination also deserve further attention. There is certainly great potential in these areas, and only a handful of papers in the literature have investigated these themes. There may also be scope for further theories from mainstream RL literature to be applied to the RL-TSC domain in the future.

References

1. Horizon 2020. URL <http://ec.europa.eu/programmes/horizon2020/en>
2. Traas: Traci as a service. URL <http://traas.sourceforge.net/cms/>
3. Abdoos, M., Mozayani, N., Bazzan, A.: Traffic light control in non-stationary environments based on multi agent q-learning. In: Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on, pp. 1580–1585 (2011). DOI 10.1109/ITSC.2011.6083114
4. Abdoos, M., Mozayani, N., Bazzan, A.: Hierarchical control of traffic signals using q-learning with tile coding. *Applied Intelligence* **40**(2), 201–213 (2014). DOI 10.1007/s10489-013-0455-3
5. Abdulhai, B., Kattan, L.: Reinforcement learning: Introduction to theory and potential for transport applications. *Canadian Journal of Civil Engineering* **30**(6), 981–991 (2003). DOI 10.1139/l03-014
6. Abdulhai, B., Pringle, R., Karakoulas, G.: Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering* **129**(3), 278–285 (2003). DOI 10.1061/(ASCE)0733-947X(2003)129:3(278)
7. Arel, I., Liu, C., Urbanik, T., Kohls, A.: Reinforcement learning-based multi-agent system for network traffic signal control. *Intelligent Transport Systems, IET* **4**(2), 128–135 (2010). DOI 10.1049/iet-its.2009.0070
8. Bakker, B.: Cooperative multi-agent reinforcement learning of traffic lights. In: ACM Trans. on Multimedia Computing, Communications, and Applications (2005)
9. Bakker, B., Whiteson, S., Kester, L., Groen, F.: Traffic light control by multiagent reinforcement learning systems. In: R. Babuka, F. Groen (eds.) *Interactive Collaborative Information Systems, Studies in Computational Intelligence*, vol. 281, pp. 475–510. Springer Berlin Heidelberg (2010). DOI 10.1007/978-3-642-11688-9_18
10. Balaji, P., German, X., Srinivasan, D.: Urban traffic signal control using reinforcement learning agents. *Intelligent Transport Systems, IET* **4**(3), 177–188 (2010). DOI 10.1049/iet-its.2009.0096
11. Bazzan, A.L.C.: Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. *Autonomous Agents and Multi-Agent Systems* **18**(3), 342–375 (2009). DOI 10.1007/s10458-008-9062-9
12. Bazzan, A.L.C., Klgl, F.: A review on agent-based technology for traffic and transportation. *The Knowledge Engineering Review* **29**, 375–403 (2014). DOI 10.1017/S0269888913000118
13. Brys, T., Pham, T.T., Taylor, M.E.: Distributed learning and multi-objectivity in traffic light control. *Connection Science* **26**(1), 65–83 (2014). DOI 10.1080/09540091.2014.885282
14. Busoniu, L., Babuka, R., Schutter, B.: Multi-agent reinforcement learning: An overview. In: D. Srinivasan, L. Jain (eds.) *Innovations in Multi-Agent Systems and Applications - 1, Studies in Computational Intelligence*, vol. 310, pp. 183–221. Springer Berlin Heidelberg (2010). DOI 10.1007/978-3-642-14435-6_7
15. Chen, B., Cheng, H.: A review of the applications of agent technology in traffic and transportation systems. *Intelligent Transportation Systems, IEEE Transactions on* **11**(2), 485–497 (2010). DOI 10.1109/TITS.2010.2048313

16. Chin, Y.K., Bolong, N., Yang, S.S., Teo, K.: Exploring q-learning optimization in traffic signal timing plan management. In: Computational Intelligence, Communication Systems and Networks (CICSyN), 2011 Third International Conference on, pp. 269–274 (2011). DOI 10.1109/CICSyN.2011.64
17. Cools, S.B., Gershenson, C., DHooghe, B.: Self-organizing traffic lights: A realistic simulation. In: M. Prokopenko (ed.) *Advances in Applied Self-organizing Systems*, Advanced Information and Knowledge Processing, pp. 41–50. Springer London (2008). DOI 10.1007/978-1-84628-982-8_3
18. Dresner, K., Stone, P.: Multiagent traffic management: Opportunities for multiagent learning. In: K. Tuyls, P. Hoen, K. Verbeeck, S. Sen (eds.) *Learning and Adaption in Multi-Agent Systems, Lecture Notes in Computer Science*, vol. 3898, pp. 129–138. Springer Berlin Heidelberg (2006). DOI 10.1007/11691839_7
19. El-Tantawy, S., Abdulhai, B.: An agent-based learning towards decentralized and coordinated traffic signal control. In: Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on, pp. 665–670 (2010). DOI 10.1109/ITSC.2010.5625066
20. El-Tantawy, S., Abdulhai, B.: Multi-agent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc). In: Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on, pp. 319–326 (2012). DOI 10.1109/ITSC.2012.6338707
21. El-Tantawy, S., Abdulhai, B., Abdelgawad, H.: Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): Methodology and large-scale application on downtown toronto. *Intelligent Transportation Systems, IEEE Transactions on* **14**(3), 1140–1150 (2013). DOI 10.1109/TITS.2013.2255286
22. Houli, D., Zhiheng, L., Yi, Z.: Multiobjective reinforcement learning for traffic signal control using vehicular ad hoc network. *EURASIP J. Adv. Signal Process* **2010**, 7:1–7:7 (2010). DOI 10.1155/2010/724035
23. Isa, J., Kooij, J., Koppejan, R., Kuijer, L.: Reinforcement learning of traffic light controllers adapting to accidents. In: *Design and Organisation of Autonomous Systems* (2006)
24. Khamis, M., Gomaa, W.: Enhanced multiagent multi-objective reinforcement learning for urban traffic light control. In: Machine Learning and Applications (ICMLA), 2012 11th International Conference on, vol. 1, pp. 586–591 (2012). DOI 10.1109/ICMLA.2012.108
25. Khamis, M., Gomaa, W., El-Shishiny, H.: Multi-objective traffic light control system based on bayesian probability interpretation. In: Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on, pp. 995–1000 (2012). DOI 10.1109/ITSC.2012.6338853
26. Khamis, M.A., Gomaa, W.: Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework. *Eng. Appl. Artif. Intell.* **29**, 134–151 (2014). DOI 10.1016/j.engappai.2014.01.007
27. Krajzewicz, D., Erdmann, J., Behrisch, M., Bieker, L.: Recent development and applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements* **5**(3&4), 128–138 (2012)
28. Kuyer, L., Whiteson, S., Bakker, B., Vlassis, N.: Multiagent reinforcement learning for urban traffic control using coordination graphs. In: W. Daelemans, B. Goethals, K. Morik (eds.) *Machine Learning and Knowledge Discovery in Databases, Lecture Notes in Computer Science*, vol. 5211, pp. 656–671. Springer Berlin Heidelberg (2008). DOI 10.1007/978-3-540-87479-9_61
29. Liu, Z.: A survey of intelligence methods in urban traffic signal control. *IJCSNS International Journal of Computer Science and Network Security* **7**(7) (2007)
30. Lu, S., Liu, X., Dai, S.: Incremental multistep q-learning for adaptive traffic signal control based on delay minimization strategy. In: Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on, pp. 2854–2858 (2008). DOI 10.1109/WCICA.2008.4593378
31. Nair, R., Varakantham, P., Tambe, M., Yokoo, M.: Networked distributed pomdps: A synthesis of distributed constraint optimization and pomdps. In: *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 1, AAAI'05*, pp. 133–139. AAAI Press (2005)

32. Ono, N., Fukumoto, K.: A modular approach to multi-agent reinforcement learning. In: G. Wei (ed.) *Distributed Artificial Intelligence Meets Machine Learning Learning in Multi-Agent Environments*, *Lecture Notes in Computer Science*, vol. 1221, pp. 25–39. Springer Berlin Heidelberg (1997). DOI 10.1007/3-540-62934-3_39
33. Pham, T., Brys, T., Taylor, M.E.: Learning coordinated traffic light control. In: *Proceedings of the Adaptive and Learning Agents workshop (at AAMAS-13)* (2013)
34. Prashanth, L., Bhatnagar, S.: Reinforcement learning with average cost for adaptive control of traffic lights at intersections. In: *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pp. 1640–1645 (2011). DOI 10.1109/ITSC.2011.6082823
35. Prashanth, L., Bhatnagar, S.: Reinforcement learning with function approximation for traffic signal control. *Intelligent Transportation Systems, IEEE Transactions on* **12**(2), 412–421 (2011). DOI 10.1109/TITS.2010.2091408
36. Puterman, M.L.: *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st edn. John Wiley & Sons, Inc., New York, NY, USA (1994)
37. Rummery, G.A., Niranjan, M.: On-line Q-learning using connectionist systems. Tech. Rep. 166, Cambridge University Engineering Department (1994)
38. Salkham, A., Cahill, V.: Soilse: A decentralized approach to optimization of fluctuating urban traffic using reinforcement learning. In: *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pp. 531–538 (2010). DOI 10.1109/ITSC.2010.5625145
39. Salkham, A., Cunningham, R., Garg, A., Cahill, V.: 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*, vol. 2, pp. 560–566 (2008). DOI 10.1109/WIIAT.2008.88
40. Singh, S., Jaakkola, T., Littman, M., Szepesvri, C.: Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning* **38**(3), 287–308 (2000). DOI 10.1023/A:1007678930559
41. Steingrver, M., Schouten, R., Peelen, S., Nijhuis, E., Bakker, B.: Reinforcement learning of traffic light controllers adapting to traffic congestion. In: *Proceedings of the Belgium-Netherlands Artificial Intelligence Conference, BNAIC05* (2005)
42. Sutton, R.S., Barto, A.G.: *Introduction to Reinforcement Learning*, 1st edn. MIT Press, Cambridge, MA, USA (1998)
43. Thorpe, T.L., Anderson, C.W.: Traffic light control using sarsa with three state representations. Tech. rep., IBM Corporation (1996)
44. Watkins, C., Dayan, P.: Technical note: Q-learning. *Machine Learning* **8**(3-4), 279–292 (1992). DOI 10.1023/A:1022676722315
45. Watkins, C.J.C.H.: *Learning from delayed rewards*. Ph.D. thesis, King's College, Cambridge, UK (1989)
46. Webster, F.V.: *Traffic signal settings*. Road Research Technical Paper No. 39, Road Research Laboratory, England, published by HMSO (1958)
47. Wen, K., Qu, S., Zhang, Y.: A stochastic adaptive control model for isolated intersections. In: *Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on*, pp. 2256–2260 (2007). DOI 10.1109/ROBIO.2007.4522521
48. Wiering, M.: Multi-agent reinforcement learning for traffic light control. In: *Proceedings of the Seventeenth International Conference on Machine Learning, ICML '00*, pp. 1151–1158. Morgan Kaufmann Publishers Inc., San Francisco (2000)
49. Wiering, M., van Otterlo, M. (eds.): *Reinforcement Learning: State-of-the-Art*. Springer (2012)
50. Wiering, M., Vreeken, J., van Veenen, J., Koopman, A.: Simulation and optimization of traffic in a city. In: *Intelligent Vehicles Symposium, IEEE*, pp. 453–458 (2004). DOI 10.1109/IVS.2004.1336426
51. Woolridge, M.: *Introduction to Multiagent Systems*. John Wiley & Sons, Inc., New York, NY, USA (2001)
52. Xu, L.H., Xia, X.H., Luo, Q.: The study of reinforcement learning for traffic self-adaptive control under multiagent markov game environment. *Mathematical Problems in Engineering* **2013** (2013)