Decentralized traffic management system via reinforcement learning

Samer I. Mohamed
October University for Modern Sciences and Arts (MSA)

Abstract: *As traffic increases throughout the globe, it becomes more and more apparent that the most appropriate response to its stochastic nature is real-time dynamic traffic control. While actuated systems to a large extent adjust well to traffic conditions, they cannot adjust fully to the hectic nature of traffic in real-time as while they are responsive to the presence of vehicles, they are not sensitive to the traffic demand (the number of vehicles), therefore performing poorly in saturated or nearly-saturated environments, introducing the need for a newer approach to control the problem. The purpose of this paper is to introduce a heuristic method to control the traffic problem and to prove how the Q-learning fits for purpose of handling the traffic light optimization problem. Validating the value of the proposed algorithm via set of metrics and KPIs like average delay per vehicle, and average number of stops per vehicle within a specified network to show the added value against other well know benchmarks.*

Keywords: *Reinforcement learning, Q-learning, adaptive traffic control.*

1. Introduction

Traffic has been growing exponentially over the years. It has been and is still getting increasingly problematic leading to tremendous amounts of time wasted each and every day commuting, and increased levels of pollution. A straightforward solution would be to expand roads, but as that is not always possible due to space unavailability, a solution needs to be devised to optimize traffic flow within the existing road infrastructure. The common and popular way to control intersections traffic is the traffic lights where traffic light controllers used to coordinate the time between different crossing traffic flows within an intersection. Historically, traffic light coordination was static with pre-timed traffic light signals based on traffic volume data and statistics. As traffic is not a consistent entity, this approach was only successful when traffic was not a major concern; however, in recent times, a system based only on statistics and historical data is not sufficiently effective, leading to the development of more traffic-sensitive systems. Actuated traffic controllers have been in development for decades. Their mode of operation is to continuously check traffic flow and adjust the signal accordingly. The duration spent in the intersection and the sequence of phases are calculated to minimize lost time and to increase the infrastructure utilization for a specific intersection or a network of intersections.

A few such systems are already in use, the most known of which are the Sydney Coordinated Traffic System (SCATS) [2], the Split Cycle and Offset Optimization Technique (SCOOT) [1], and the Optimized Policies for Adaptive Control (OPAC) [3]. These systems employed online optimization methods to dynamically adjust signal timings by utilizing various methods of surveillance to monitor current traffic and react accordingly. They gather information such as traffic flow, average speed, lengths of queues, numbers of vehicles waiting, and accident detection, and accordingly predict the best option suited for the situation at hand and change the traffic lights in respective areas accordingly. SCATS and SCOOT

differ from OPAC in that the former operates in a centralized manner which makes these systems suffer from certain limitations such as having a reliable communication networks to transfer the control actions from a central computer to local controllers within each intersection. In the case that the communication network fails, local controllers would be forced to switch to an offline mode and operate independently of each other. Additionally, centralized systems are not scalable to expand as they are often designed around a maximum network size which means the addition of a few controllers would require extensive upgrades. OPAC, however, is built in a decentralized fashion which makes it computationally less damaging as each local controller only maintains the relevant information from the surrounding intersections and controllers making dependability on a central computer for real-time commands non-existence and making it significantly easier to scale. However, OPAC still has its drawbacks as it's built using dynamic programming techniques to model and solve the traffic signal control problems. While dynamic programming is theoretically ideal as it perfectly captures the stochastic nature and dynamics of a traffic system, it, practically, requires a model for the environment. Also as the number of states that could represent wide traffic conditions is typically massive, and since dynamic programming algorithms are computationally intractable, it hindered the straightforward network-wide application.

Thus the need to start basing traffic control systems on artificial intelligence adopting reinforcement learning principles which have been proven effective in different areas of traffic control and more efficient for traffic signal optimization than any existing traditional approaches. In this paper, traffic signal optimization is implemented and tested using Q-learning learning algorithm. The results and effectiveness of the algorithm are proved with set of KPIs and measures like the average delay per vehicle, and the average number of stops per vehicle.

2. Background

A. Fixed-time control

Timing plan for the traffic signal is developed off-line based on historical data, where the cycle and offset times are optimized off-line before being deployed on the environment. Thus the duration and phases remain fixed and don't adapt to environment dynamics of the traffic demand. Implementation costs for those systems are relatively low compared to others due to no need to sensors or loop detectors to sense the environment or traffic demand. Thus best fits for those areas where traffic dynamics are low and pretty predictable [5].

B. Actuated control

Unlike pre-timed signal control, actuated traffic signal control systems are responsive to traffic flow fluctuations. Actuated traffic control requires actuated traffic controllers and vehicle detectors placed on the approaches of the intersection. Actuated control systems use sensors or detectors at the intersection to detect any vehicle at the intersection. Different types of actuated systems like Fully-actuated signals that detectors on all of the approaches and semi-actuated signals that have detectors only at some of the approaches. The detector controls the phased sequence based on the number of vehicles at the intersection approaches. Each and every phase is given enough time function in the number of vehicles at the intersection. This can be extended if more vehicles are detected during the green light. This additional time or 'passage time' will be added to the phase time up to some set of maximum green time. This phase sequence will be managed by the traffic controller dependent on the dynamics at the intersection itself [6].

C. Adaptive control

Adaptive signal control is the newly adopted methodology for current traffic systems where signal

timing actions are continuously updated according to traffic dynamics and the number of vehicles approaching the intersection. Adaptive control system parameters like cycle time, offset time, and split time are calculated towards a specific objective function, and then the traffic controller automatically executes these parameters via set of actions in response to the real-time traffic dynamics [7]. Reinforcement learning is considered one of the main and basic recent fourth generation adaptive traffic management system control which consists of:

- Environment states (S)
- Agent actions (A)
- State transition rules (T)
- Reward function (R)

The basic concept of reinforcement learning [10] designed around an agent that observes its surrounding environment, and then takes actions that changes the state of its environment based on their observation of it, through which the agent either receives a reward or a penalty for their decision. Based on the rewards received, the agent learns and starts taking decisions to maximize the objective or utility function. Reinforcement learning is one of these learning algorithms that usually works on a discrete time steps, where in each time step (t), the agent observes their environment and deduces the state (s_t). Based on that state, the agent takes an action (a_t). In the next time step, the agent gets reward (r_t). The agent then moves on to state (s_{t+1}) where it repeats the process.

Reinforcement learning includes three different methods of solving problems; Dynamic Programming (DP) [10], Temporal Difference (TD) [10] and the Monte Carlo methods (MC) [10]. The proposed algorithm is based on Q-learning as an incremental reinforcement learning algorithm based on TD to find an optimal action-selection policy. In Q-learning, as detailed in the following section, the agent doesn't require a model and learns from its experiences instead through continuous utility function updates. The utility function is updated at every time-step based on the reward received for the action taken at the previous time-step as shown in figure 1.

Q-Learning [10] classified into two main categories, model-based and model-free learning algorithm. For model-free reinforcement learning technique, the agent learns from its actions until it converges to a specific objective function or optimal policy, after which, the optimal policy can be executed by selecting the action with the highest expected value in each state. While model-based approach uses a specific model that maps the states and actions towards the optimal policy selection which makes it perfect to handle stochastic environments such as that of traffic without requiring any adaptations. At every time-step, the algorithm makes the following update:

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha\{r_{t+1} + \gamma \, max \, Q(s_{t+1}, a_{t+1}) - Q(s_t,a_t)] \quad (1)$$

where (α) is the learning rate, (γ) is the future reward discount factor, and $Q(s_t,a_t)$ is the value function of the state (s_t) paired with action (a_t)

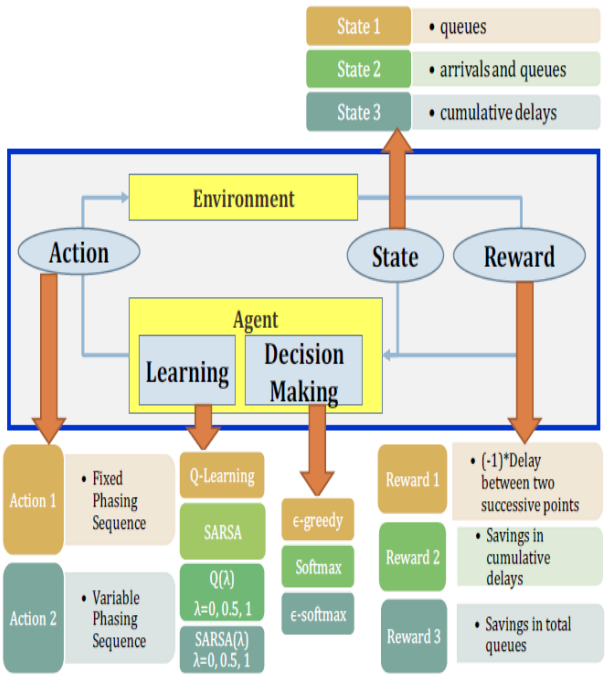


Figure 1: Reinforcement learning structure

In order to converge to the optimal policy in a stochastic environment like the given one, the learning rate (α) must be reduced over the learning time steps.

The learning rate decides how much newly acquired data affects the learning process and the policy. With learning rate set at 1, the agent will only consider the newest information, and for a learning rate of 0 means that the agent will not learn at all. The learning rate should be set relatively low so that every new experience and state the agent goes through affects the learning process and policy marginally as opposed to the agent learning based on the newest experiences to a larger extent than it should. The discount factor ***(γ)*** decides whether actions should be taken leaning towards the short-term or long-term goals as it defines the severity of the future rewards or penalties. A discount factor of 0 means the agent will only take the immediate reward received into consideration ignoring the long-term rewards and therefore goals, while a discount factor of 1 means the agent will consider the long-term rewards. Research and previous studies have shown that using a high discount factor (closer to 1) always got better results than using one closer to 0. The analysis for the proposed system proves that using a discount factor of **(0.6)** was optimal to achieve the best outcomes.

Multi Agent Reinforcement Learning for Integrated Network (MARLIN) [8] is considered one of the Adaptive Traffic System Control (ATSC) systems that is designed as a one of the Intelligent Transportation Systems (ITS) to increase the infrastructure utilization efficiency, manage demand. MARLIN's success is largely a result of its adaptive nature, Agents in MARLIN learn together towards the final objective and not in isolation like other MARL algorithms. It enables traffic lights or agents to self-learn and collaborate with neighbouring traffic lights to improve trip duration, minimize delays, pollution, and fuel consumption as shown in figure 2.

3. The proposed framework architecture

Since all existing adaptive traffic systems are implemented on a centralized architectural model, they have a single point of failure; if the central computer fails, coordination between the intersections will be lost. The proposed system is decentralized keeping each intersection independent and the communication between them is well protected. Most systems rely on inductive loop sensors to observe the environment through detecting the vehicles in each lane to measure the queue lengths in the intersections. While inductive loops [11] have their merits, they also have their drawbacks; their installation is costly as it's done underground and requires digging (which means that the roads would have to be closed, leading to an increase in the traffic where system is trying to prevent).

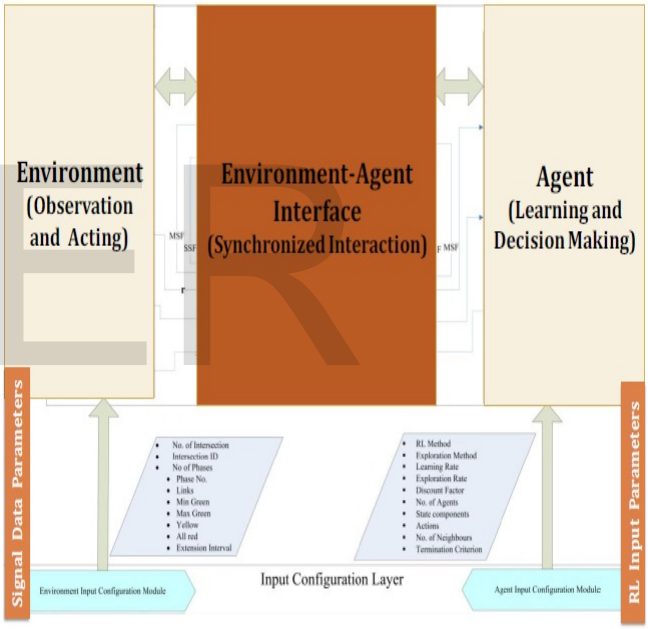


Figure 2: Adaptive Traffic Signal Control structure

Wire loops [12] are subject to wear and tear because of the weather and the pressure applied to them throughout the day leading to the need for regular maintenance. In the proposed system, ultrasonic sensors is selected because of its reliability and better communication quality.

A. Ultrasonic sensors

Ultrasonic sensors [13] have many advantages that distinguishes them from inductive loops; they are installed above ground on traffic signals

themselves minimizing installation costs, they are highly accurate (85%) accuracy and (97%) when two sensors are installed in succession, and additionally they can monitor multiple lanes. Through the proposed system, ultrasonic sensors are utilized to detect vehicle velocity, vehicle count and vehicle pattern/shape. In the proposed system, two sensors are separated by a known distance (1.6 m), the first sensor starts a timer when a vehicle enters its range and the second sensor stops the timer when the vehicle enters its range. As the distance and the time taken for the vehicle to enter the range of the first sensor and range of the second sensor are known now, the velocity of the vehicle can be calculated. It also counts the numbers of vehicles and identify the type of the vehicle whether it is a passenger vehicle or a van, using the designed algorithm to recognize the pattern of the vehicle.

We proposed a vehicle pattern algorithm where two sensors being fixed parallel to the ground, so that the waves generated is perpendicular to the ground. Ultrasonic sensor only receive waves reflected normal to the ground and normal to the sensor, any waves reflected with angle will be not seen by the sensor. This fact is used to identify the vehicle type, because the front and rear glass of the vehicle are tilted by an angle, though the waves reflected from them will not be seen by the sensor and this will create a drop in the signal coming from the sensor. The proposed algorithm is designed to trace these drops and count them and identify the vehicle type based on this criteria. For example Passenger type vehicles will give 2 drops in signal and SUV vehicles will give only one drop, buses on the other hand will give no drops in signal. The challenge with this approach is related to the fact of how to differentiate between van type vehicles and SUV, both will give one drop in signal, and there are some other cases with the same conditions, Thus the proposed algorithm mitigate this by using the length of the vehicle as a parameter to help in distinguish between those cases. Although the challenge to calculate the vehicle length from only knowing it's speed, the proposed algorithm utilize the two sensors fixed on the environment while vehicle is passing through them to calculate the time it takes while is function in the vehicle speed and length. With these parameters the proposed algorithm successfully distinguish between vehicles and identify its type [14].

B. Communication system

The communication between the different sensors and the agent is done using ZigBee wireless [11] units to transfer the state's data from the sensor to the agent where the core process will run its calculations and send the output back to the traffic signal to either extend the current phase or terminate it. The core agent has its ZigBee as a coordinator to receive data from other sensors which their ZigBee's are set as routers to send data to the core agent. The proposed algorithm is connecting the ZigBee wireless module to the ultrasonic sensor where for each agent the ability to transfer the data between the sensors and themselves. The system consists of one ZigBee set as a coordinator (main agent) and the other ZigBee set as routers. The routers always send data to the coordinator to process it and then based on the core learning algorithm it takes actions.

C. Core algorithm

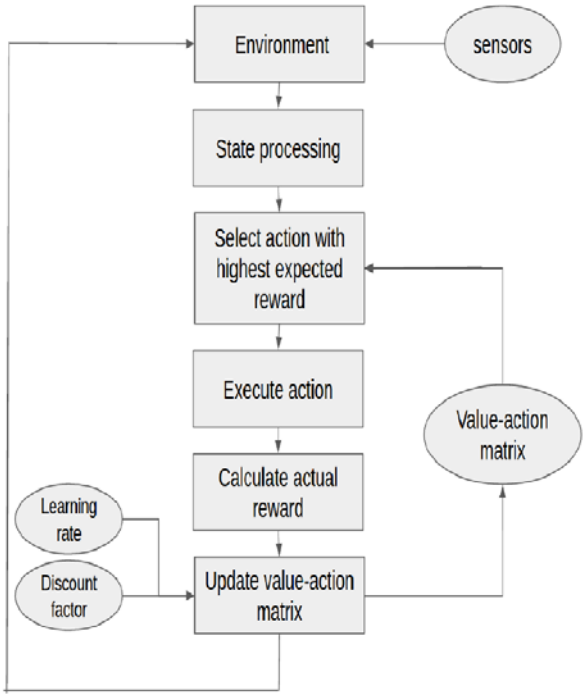
Our proposed system is based on Q-learning algorithm for the main core learning system architecture. The core algorithm is implemented on a single agent operating in a single intersection. The network consists of an intersection of two streets with turning lanes. Each street in the intersection is controlled by a traffic light that had two phases, green, and red. Q-learning algorithm found to be the most suitable for our proposed traffic system as it suits the stochastic nature of traffic as detailed in the previous sections. The algorithm is an off-policy one where a value-action matrix is updated

at each transition between states. Through this state transition, an action is chosen, a reward is assigned then matrix is updated based on the reward that action yielded and on factors such as the learning rate, and the discount factor. This discount factor controls the rate by which the agent learns and whether it puts more emphasis on the short or long term objectives. As the algorithm is in use for longer and longer periods of time, it learns more from its actions and observations of its environment until it converges onto an optimal policy to follow.

Figure 3 shows the flow chart of the proposed core algorithm which starts by collecting the environment/street data via sensors to measure the traffic demand at any point of time and use this information to build the state vectors/matrix. Algorithm then selects the best action from the Q-table based on the state vector to achieve the maximum reward towards the final objective. After selection the action (extend green or terminate current phase), the algorithm execute the action and update the Q-table to reflect the learning from executing the action. Based on the actual action, the system will move to next state, thus system updating the state/action pair table and reiterate this process till achieve the final objective based on specific learning rate **(α)** and discount factor (***γ)***.

There were two actions, (1) extend green, and (2) switch to red. The durations of time signals were set to a constant where each phase would last 8 seconds, after which, it would either get extended or terminated. As the durations were constant, the duration of the signal is not included in the state definition. The queues are split on 10 classes which defined at intervals of 5 vehicles, and 10, where velocity splits at intervals of 6 Km/s ex: Q=5 and Speed=6s composes the first state, Q=5 and speed =12 composes the second state and so on), therefore the number of all possible states is a 100 and the number of all possible state-action pairs is 200 which proved to be sufficient to reflect the environment.

At every time step, the agent receives the states from the sensors at the intersection, and chooses one of the two possible actions, whether to extend the green phase, or to switch to red, at the beginning of the next time step, before another action is taken, and based on the new information received, the agent either receives a reward or a penalty for the action taken at the previous time-step. Several reward functions were considered and tested such as the difference in average delay, the difference between the number of vehicles that left the network and those remaining, the difference in average velocity, among many others, but ultimately testing proved the difference in queue length to yield the best results when it comes to the final objective of the agent. There are different objectives tested throughout the system testing phase like minimizing delay, minimizing average number of stops per vehicle, minimizing queue lengths, and maximizing average velocity like



detailed in the next section.

Figure. 3: Proposed core algorithm flow chart

4. Testing and evaluation

A. Testing strategy

The proposed framework for the traffic signal optimization is tested for all different combinations of traffic system parameters, along with different state definitions, different reward functions, and different action-selection functions. Initially the core algorithm was tested using 6 reward functions, The 3 that yield the best results with respect to the average delay:

- Difference in queue
- Queue/capacity
- Difference between vehicles that cleared the intersection and remaining ones)

The two combinations that yield the best results were tested individually afterwards using 10 different values for the learning rate between [0, 1] and 10 different values for the discount factor, also between [0, 1]. Additionally, 2 different learning rate values were selected, one of which used an initial learning rate value that was updated with very small increments, and the other was a function of the number of visits to the state-action pair at hand. Traffic volume (number of vehicles per hour) was randomized at every time step to reflect the stochastic nature of arrivals in a real road network. All testing was done using the software PTV VISSIM [12]. The core proposed algorithm was written in C++ and communicated with PTV VISSIM (which simulates the environment streets and traffic dynamics) via its COM interface (Which simulates the traffic agent which controls the intersections signals and control the traffic in the streets based on the gathered data from the VISSIM simulator). After checking the simulation results compared to the actual results of other simulators, the simulator's accuracy is about 97% form the real world witch is considered very high compared to other simulation programs and also to ensure the accuracy of achieved results, the simulation run repeated 3 times and took average reading.

Testing mechanism divided into two main components, testing the hardware/communication subsystem and testing the core software intelligent algorithm. Testing the hardware subsystem started by unit testing each one of the ultrasonic sensors using a vehicle to evaluate the sensor behavior towards it. This requires testing the data transmission process by making one sensor acts as a coordinator and the other acting as a router watching the data flow into the system serial ports. All other sensor boards are tested with the main coordinator, to ensure that every sensor is working probably and sending accurate data. This followed by testing the whole sensors together with coordinator to measure the reliability of the whole system.

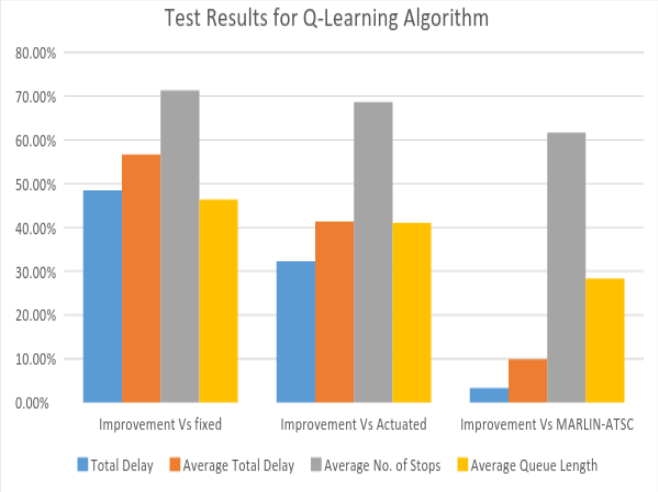
Sensors testing is done in a garage with a height of 3 meters where one sensor is fixed at the ceiling of the garage heading downwards along with other 2 ultrasonic sensors with a separation of 1.6 meters between them. Measurement of the velocity and the vehicle type is done first on only one vehicle with different speeds, then tried the same testing approach on couple of other vehicle types to ensure that the counter and the average speed is working probably. Communication between the sensors and agents is tested first wit wired connection via a USB cable connected to a laptop then tested the wireless communication via the ZigBee protocol as the main communication channel.

B. Results analysis and evaluation

The proposed system results extracted based on hundreds of test runs, these results are measured against VISSIM's embedded data when operating independent of our algorithm. These results evaluated against average delay, average velocity, and average number of stops per vehicle.

Table 1. Test results for proposed system architecture

All measurements for total delay and average delay and average queue length were done in two rush hours

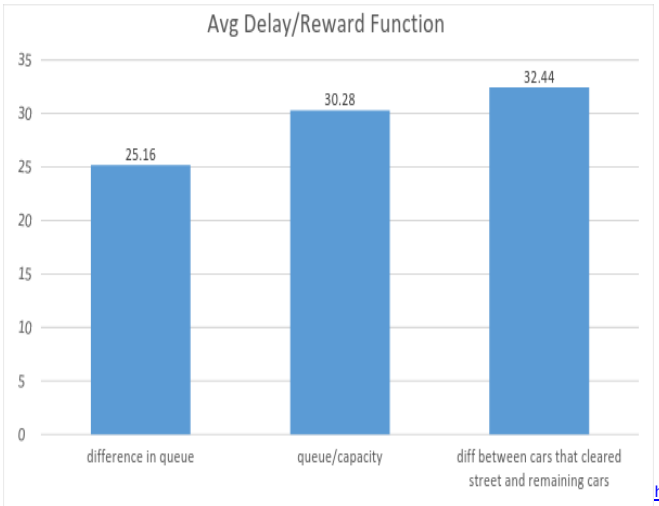


[14:00 →16:00] with maximum number of vehicles (2 input vehicles per second) in the same network (one 4-roads intersection) that matches those for the other bench marks in this field (Fixed, actuated, MARLIN) to be able to compare the proposed system results and show the achieved improvements as detailed in Table 1 and figure 5.

Figure 4: Proposed system test results

These results were achieved under the following parameters:

- Reward Function: Difference in queue
- Discount Factor: 0.6
- Learning Rate = 1 / v(s,a)
- ε-greedy
- State Definition: Queue length and average velocity



Control System	Total Delay	Average Total Delay	Average No. of Stops	Average Queue Length
Fixed Time	246085	58.05	1.88	17.55
Actuated	187214	42.90	1.72	15.96
MARLIN-ATSC	122620.5	27.92	1.41	13.13
Proposed algorithm	126716.2	25.16	0.54	9.41
% Improvement vs fixed	48.51%	56.66%	71.28%	46.38%
% Improvement vs	32.31%	41.35%	68.6%	41.04%

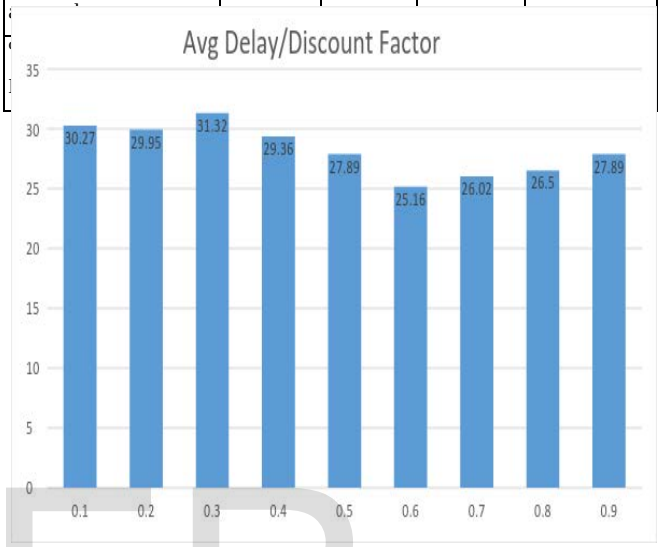
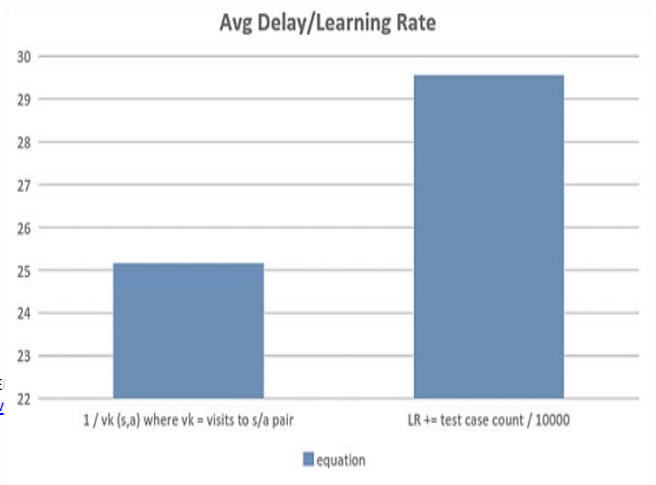


Figure 5: Proposed core algorithm reward function impact on average delay

The results clearly show a remarkable improvements over fixed-time and actuated systems in all comparison fields, and a moderate achievement in decreasing delay over MARLIN-ATSC with a great improvement in average number of stops and queue lengths. The results achieved with different reward functions are summarized in figure 5, discount factors as shown in figure 6 and learning rates as indicated in figure 7.

Figure 6: Proposed core algorithm Discount factor value impact on



average delay

Figure 6 shows that the minimum delay achieved when using the 'Difference in queue' reward function compared to the other reward functions with around 15% less compared to 'queue capacity' and 20% less compared to 'Difference between vehicles that clear streets and those remaining'.

Figure. 7: Proposed core algorithm learning rate value impact on average delay

Using discount factor of 0.6 prove to be the best value that achieved the minimum average delay compared to the other values between [0, 1] as shown in figure 6. When using the equation of number of visits to the (State/Action) or (S,A) pair for learning approach the proposed algorithm achieve less average delay of 15% compared to the same one when using number of iterations/test cases count as shown in figure 7.

5. Conclusions

Adaptive Traffic Signal Control systems proves to be one of the most current methodologies that used to adapt with traffic dynamics and fluctuations in an effective way. Reinforcement learning is one of the learning algorithms used for these adaptive traffic systems that offers significant advantages in the application to real-time traffic system. Adopting model-free learning proves to outperform those model-based or even those systems depend on traffic stats for traffic system management. Our proposed system based on Q-Learning algorithm build its learning experience through dynamic interaction with the environment rather than rely on pre-specified models of these processes. The results achieved from the proposed algorithm shows remarkable improvements against different KPIs/bench marks. Future work of the proposed system is to scale it and apply on network scale to implement on more than one intersection to achieve multiple objectives concurrently. Although achieving multiple objectives on the network level is much harder than focusing on one intersection because to understand the effect of every decision made by the agent on the network, the expected results looks very promising

References

[1] SCOOT - The world's leading adaptive traffic control system. (n.d.). Retrieved from http://www.scoot-utc.com.
[2] Performance you can see. (n.d.). Retrieved from http://w3.usa.siemens.com/mobility/us/en/urban-mobility/road-solutions/adaptive-software/Pages/scoot.aspx.
[3] P B Hunt, D I Robertson and R I Winton. "SCOOT – a traffic responsive method of co-ordinating signals." TRL La-boratory Report 1014 (1981).
[4] R D Bretherton and G T Bowen. "Latest Developments in SCOOT – SCOOT version 3.1"Proceedings IEE 6th Inter-national Conference on Road Traffic Control. London (April 1996).
[5] R D Bretherton, K Wood and G T Bowen. "SCOOT Version 4." Proceedings IEE 9th International Conference on Traffic monitoring and control. London (April 1998).
[6] R D Bretherton, M Bodger and N Baber. "SCOOT – Managing Congestion Communications and Control." Proceed-ings of ITS World Congress. San Francisco (2005).
[7] Colyer. "SCOOT from Scratch – Experience in Worcester." Proceedings of PTRC Summer Annual Meeting. University of Sussex. (July 1985).
[8] El-Tantawy, S. (2012, November/December). Multi-agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC) [Scholarly project]. In TSpace. Retrieved from https://tspace.library.utoronto.ca/handle/1807/67269.
[9] A Summary of Vehicle Detection and Surveillance Technologies used in Intelligent Transportation Systems. (2007, August 31). From http://www.fhwa.dot.gov/policyinformation/pubs/vdstits2007/vdstits2007.pdf.
[10] Silver, D. (n.d.). Reinforcement Learning. Lecture. Retrieved from http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html (Aug 2015).
[11] Sunkari, S., Parker, R., Charara, H., & Palekar, T. (2005). EVALUATION OF COST-EFFECTIVE TECHNOLOGIES FOR DVANCE DETECTION. Texas: Texas Transportation Institute The Texas A&M University System.
[12] Jo, Y., & Jung, I. (2014). Analysis of Vehicle Detection with WSN-Based Ultrasonic Sensors. doi:10.3390.
[13] Abdulhai, B. and L. Kattan (2003), Reinforcement learning: Introduction to theory and potential for transport applications, Canadian Journal of Civil Engineering, vol. 30, no. 6, pp. 981–991.
[14] N. H. Gartner, Ch. Stamatiadis and P. J. Tarnoff, "Development of Advanced Traffic Signal Control Strategies for ITS: A Multi-Level Design", Transportation Research Record 1494, TRB, pp.98-105, 1995.
[15] Sarma, D. (1996). DLLs for Beginners. Microsoft Developer Support.
[16] PTV VISSIM 8 USER MANUAL. (2015). Karlsruhe: PTV AG.
[17] PTV VISSIM 8 INTRODUCTION TO THE COM API. (2015). Karlsruhe: PTV.
[18] T. Thorpe, "Vehicle traffic light control using sarsa," Master's Project Report, Computer Science
[1]