Applying Hierarchal Clusters on Deep Reinforcement Learning Controlled Traffic Network

Fady Taher*, Ayman EL-Sayed*, Ahmed Shouman*, Ahmed El-Mahalawy*

* Dept. of Computer Science and Eng., Faculty of Elect., Eng., Minufiya University

(Received: xx-Month-201x – Accepted: yy- Month-201x)

Abstract

Traffic congestions is a crucial problem affecting cities around the globe and they are only getting worse as the number of vehicles tends to increase significantly. Traffic signal controllers are considered as the most important mechanism to control traffic, specifically at intersections, the field of Machine Learning introduces advanced techniques which can be applied to provide more flexibility and adaptiveness to traffic control techniques. Efficient traffic controllers can be designed using a reinforcement learning (RL) approach but major problems of following RL approach are, exponential growth in the state and action spaces and the need for coordination. We use real traffic data of 65 intersection of the city of Ottawa to build our simulations and show that, clustering the network using hierarchal techniques has a great potential in reducing the state-action pair significantly and enhance overall traffic performance.

Keyword: Adaptive traffic signal control, Clustering, Deep reinforcement learning, Multi-agent system, Simulation, Traffic controller.

1. Introduction

Most traffic light controllers work with far less information about the traffic, they follow a fixed protocol, that is, the light is red for some time and green for some subsequent time. The time intervals usually change during rush hours but still static.

Researchers have been trying to implement intelligent systems as a replacement for static ones to increase the efficiency of controllers, such systems use different machine learning techniques to enable signal controllers to adapt and behave based on the traffic state, this comes with a cost; as deploying such adaptive controllers at intersections without proper planning could lead to limit their potential benefits moreover, it might decrease the overall performance of the network. Therefore, optimally controlling and coordinating the operation of multiple signal controllers simultaneously is required. However, this integration adds complexity to the system.

In reinforcement learning, agents learn to map actions to states in order to maximize a numerical reward [2].

Two of the major challenges associated with implementing intelligent controllers using reinforcement learning is the need for coordination and the curse of dimensionality [3]. To address these limitations, we present a new method which uses hierarchal clustering to solve state-space problem instead of traditionally used geographical attributes and deep reinforcement learning controllers to manage vehicles flow instead of static ones, we also run our simulations based on real data covering about 50 intersection instead of a small network of 4 or 8 theoretical intersections.

The paper illustrates the approach covering clustering and Deep Reinforcement Learning in section 2. Section 3 explores contributions and similar work of other researchers. Section 4 illustrates the proposed algorithm. Results are presented with SUMO simulator [4] using real traffic data acquired from the city of Ottawa and the results are shown in section 5. Finally, section 6 presents conclusion and future work.

2. Literature review

2.1. Clustering

Clustering is categorization of items, it has been a focus of study in many fields by researchers; this reflects its importance and usefulness in exploratory data analysis. It is used in decision-making, and machine-learning like document retrieval, and image segmentation.

In this paper we will be focusing on and using hierarchical clustering technique which can be addressed by algorithm known as complete linkage.

In complete link algorithm, the distance between two clusters is determined by those two elements (one in each cluster) that are farthest from each other as stated in Eq1. Two clusters are merged to form a larger cluster based on distance criteria or threshold, figure 1 depicts an

$$D(X,Y) = \max_{x \in X, y \in Y} d(x,y) \quad (1)$$

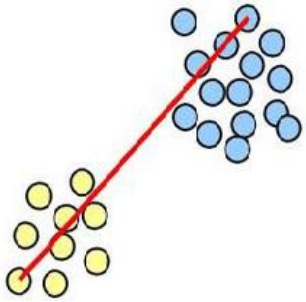

Figure 1 Complete linkage

illustrative image for complete linkage demonstration.

2.2. Deep reinforcement learning

Reinforcement Learning is the process of acquiring new knowledge and the discovery of new facts and theories through observation and experimentation.

Each time the agent performs an action in its environment, we provide the agent a reward or a penalty to indicate the benefit and importance of the resulting state. For example, when training an agent to play a game the trainer might provide a positive reward when the game is won, negative reward when it is lost, and zero reward in all other states. [5]

The Q-learning agent learns optimal mapping between the environment's state (s) and the corresponding optimal control action (a) based on accumulating rewards $r(s,a)$. In each iteration (k), the agent observes current state (s) and picks and executes action (a) that belongs to the available set of actions A; then, the Q-factor is updated according to the immediate reward $r(s,a)$ and the state transition to state(ś)as follows [6]:

$$Q^k(s^k,a^k) = (1-\alpha)Q^{k-1}(s^k,a^k) + \alpha\left[r(s^k,a^k) + \gamma \max_{a^{k+1}\in A} Q(s^{k-1},a^{k-1})\right] \quad (2)$$

Where α and γ $\in[0,1]$ are referred to as the learning rate and the discount rate, respectively. The agent can simply choose the greedy action at each iteration based on the stored Q-factors, as follows:

$$a^{k+1} \in \arg\max_{a\in A}\left[Q(s,a)\right] \quad (1)$$

However, sequence Q^K is proven to converge to the optimal value only if the agent visits the state–action pair for an infinite number of iterations [7]. This means that the agent must sometimes explore (try random actions) rather than exploit the best-known actions

In our approach, we use a deep convolutional neural network to approximate the optimal action-value function, figure 2 show a building blocks demonstration.

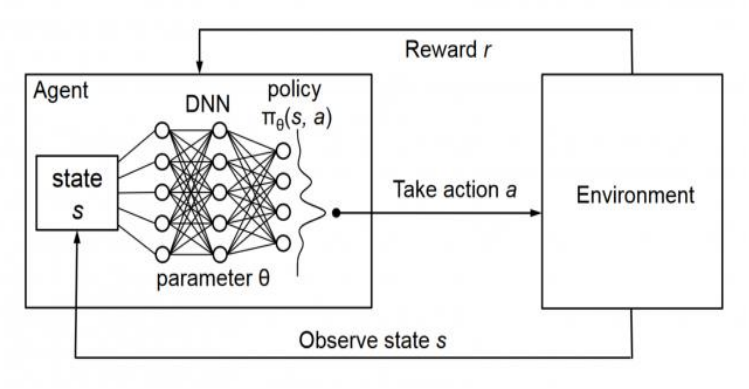

Figure 2 Deep Reinforcement Learning process

3. Related Work

Intelligent agents interact in a cooperative environment where they learn by sharing information and trial and error.

A major problem of reinforcement learning approach is the exponential growth in the state-action space.

During the past years researchers have been trying to change the way traffic controllers are operated, from using simple static controllers to adaptive controllers like actuated controllers [8] to using Machine Learning (ML) techniques, following we mention a few.

1. SS Mousavi, M Schukat applied deep reinforcement learning algorithms with focusing on both policy and value-function based methods to traffic signal control problem in order to find optimal control policies of signaling, by using raw visual input data of the traffic simulator snapshots and an agent per intersection. The approach led to promising results and showed they could find more stable control policies compared to previous work of using deep reinforcement learning in traffic light optimization [9].

2. Yilun Lin, Xingyuan Dai, et al also proposed DRL (Deep Reinforcement Learning) dedicated to large-scale UTC (Urban Traffic

Control)problems to learn the relationship between the states and the actions. They tested different reward functions and designed a hybrid reward, in which the throughput of the traffic network, along with the balance of queueing length around intersections is chosen as the performance indexes, they also used an agent per intersection. Tests showed that this new model could be optimized within an acceptable time for a traffic grid [10].

3. Van der Pol, Elise, and Frans A. presented a paper in which they used DQN algorithm with transfer planning and using image-like state representation and single agent per intersection, it showed a promising and scalable multi-agent approach to deep reinforcement learning, but the DQN algorithm may oscillate during training, a problem also found in earlier work on deep reinforcement learning. [11]

4. Genders, Wade, and Saiedeh Razavi applied modern deep reinforcement learning methods to build an adaptive traffic signal control agent. They proposed a state space, the discrete traffic state encoding, trained single agent per intersection using Q-learning with experience replay. Their agent was compared against a one hidden layer neural network traffic signal control agent and it managed to reduce average cumulative delay by 82%, average queue length by 66% and average travel time by 20%.[12].

4. Proposed algorithm

We propose to cluster the network using hierarchal clustering techniques; each cluster will be controlled by an agent which will be trained to control the network using deep reinforcement learning. Clustering the network will reduce state-action space and learning time. Figures 3, 4 present the difference between the currently used model and the proposed model respectively.

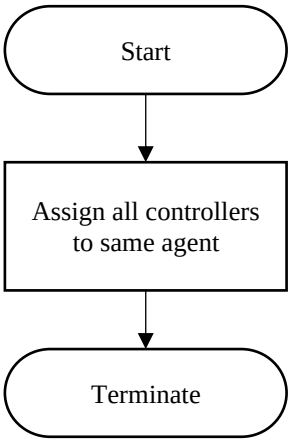

Figure 3 Current model

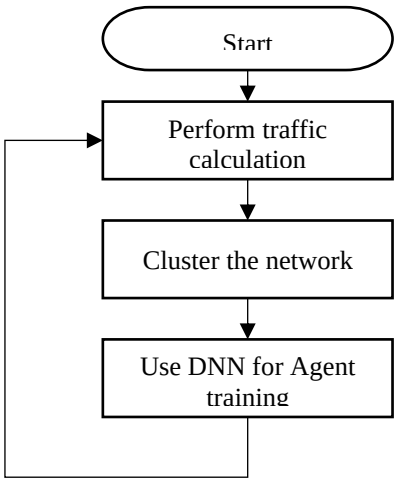

Figure 4 Proposed model

In order to cluster the network using hierarchal clustering algorithm (1) was used.

Algorithm 1 clustering TLS

1: **given**
2: `Network average traffic` $threshold$
3: A set X of agents $\{x_1, ..., x_n\}$
4: A distance function dist (c_1, c_2)
5: **for** i=1 `to` n **do**
6: c_i= $\{x_i\}$
7: **end for**
8: $C = \{c_1, ..., c_n\}$
9: **while** $size(C) > 1$ **do**
10: (c_{min1}, c_{min2}) = `minimum dist`(c_i, c_j) `for all` c_i, c_j `in` C and $\leq threshold$
11: `remove` c_{min1}, c_{min2} `from` C
12: `add` $\{c_{min1}, c_{min2}\}$ `to` C
13: **end while**
14: `train agents`

After performing 8 simulations, we obtained traffic counts between all junctions and used them to construct a distance matrix to find how “close” each junction to another we then used averaged traffic counts between pair

of junctions as threshold and implemented complete linkage to acquire final clusters.

5. Simulation setup

To demonstrate the effectiveness of intelligent traffic control system, it should be tested on realistic traffic scenarios. For this reason, a realistic traffic model based on a section of the downtown area of the City of Ottawa was simulated using SUMO simulator which was chosen for many reasons including portability, presence of an active development community and availability of a graphical user interface.

The area addressed is a 9x7 block of down-town Ottawa, with over 50 intersections requiring control. This is not an extremely large network but, it is much larger than most of the simple networks used in previous intelligent traffic signal research. This area also contains a wide range of street types ranging from small one-way residential streets with low traffic volumes, to main streets of high volume and multiple lanes.

The simulation network was acquired from the work presented by F.Taher et.al [13]. Figure 5 depicts sample intersection as seen in the simulator.

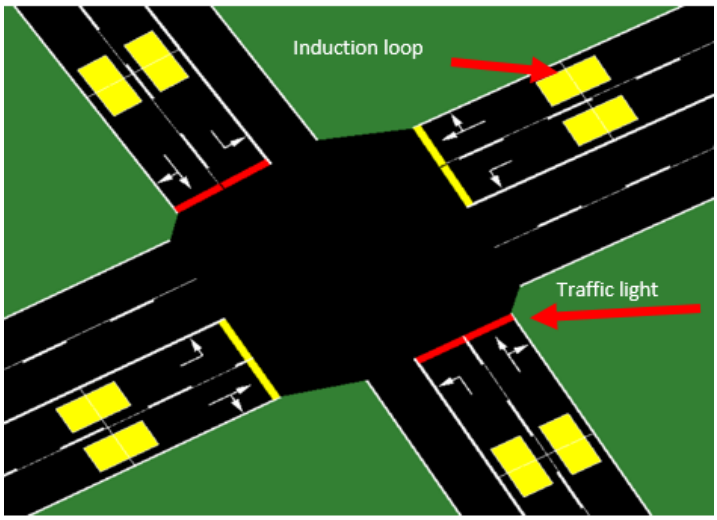

Figure 5 Network in SUMO simulator

For training step, usually Q Learning uses online learning, as in every step of the simulation a reward is received and instantly used to improve the learner. This approach is not ideal for neural networks. Because of that the idea of 'Experience Replay' is introduced. Past experiences are stored in memory and in each step, a random batch is retrieved to train the deep neural network to prevent overfitting. In addition an exploration factor is used to take random actions based on a decaying probability.

Figure 6 depicts the main program loop. Every agent chooses an action and sets it in the simulation environment. Afterwards, one simulation step is executed. Finally every agent can gather the rewards and make observations for every junction it controls. The benefits of this approach are that every agent can learn many times for every simulation step and less memory is needed as opposed to a "one agent per intesection"-approach.

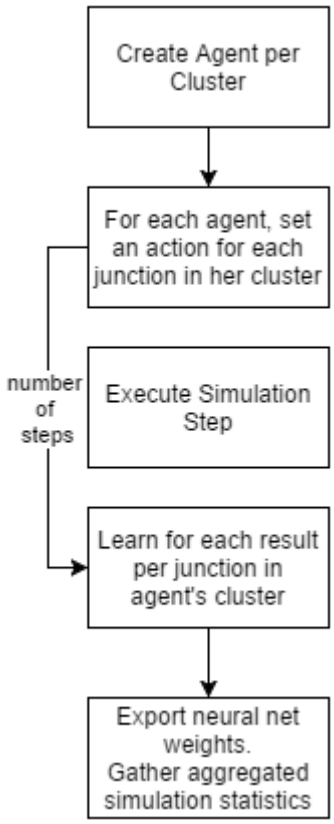

Figure 6 main program loop

5.1 State space

The first idea for the state space is to be presented using the following for each induction loop:

- The mean speed of vehicles that passed across induction loop within the last simulation step [m/s]
- The number of vehicles that were on the named induction loop within the last simulation step [#]

5.2 Action space

The action space for this particular problem gets very large very quickly due to the exponential growth. One traffic light can have the following

states, the state must be one of [rRgGyYoOu] for red, red-yellow, green, yellow, off, where lower case letters mean that the stream has to decelerate. The offline state is omitted.
Therefore, the action space size is m^n with **m** being number of possible states in this case (4) one for red, red-yellow, green, yellow and **n** being the number of traffic lights.

This introduced a very huge action space which couldn't be handled with the used hardware, a more rational approach was used, a predefined set of phases per intersection was used instead, for example an intersection could have set of 4 actions/phases like (rrrGGGG, rrryyyy, GGGrrrr, yyyrrrr) and the agent should choose to apply one of those 4.

5.3 Evaluation metrics

Two values were considered to measure the network performance:

- Average number of departed cars that is the total number of cars which have reached their destination,
- Mean Travel Time that is the average travelling time for all cars since entering the network until reaching their destination.

6. Results

To test the capability of the proposed algorithm to effectively reduce exponential growth in the state-action space, average of 8 simulation scenario were generated from the available data, each of which represents realistic vehicle volumes for an 11-hour period (7a.m.–6p.m.).

The average traffic count calculated between intersections was used as a threshold to perform the clustering algorithm, resulting in the cluster dendrogram shown in figure 7.

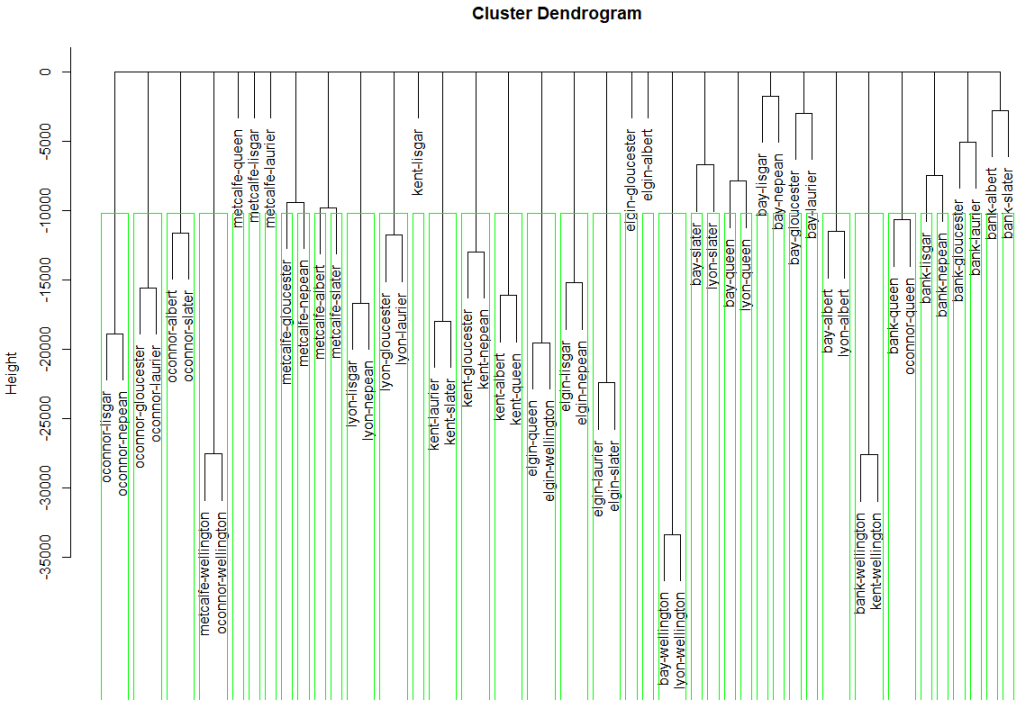

Figure 7 Complete link clustering dendrogram

Each cluster (junctions belonging to same group) are bounded by a green rectangle using complete link algorithms. The clustering pattern for complete linkage distance tends to create compact clusters, also shows that, intersections are grouped in a cluster of smaller size (1-2).

In figure 8 we can see the network junctions outlined on X and Y axes before clustering, all intersections are considered as one huge cluster network and controlled by one agent, such huge network will increase the DRL state-action space dramatically and increase learning time as a result. In fact this would have resulted in $\mathbf{3.7}e^{\mathbf{24}}$ possible action to be considered each simulation step and would be hard if not impossible to be handled in memory.

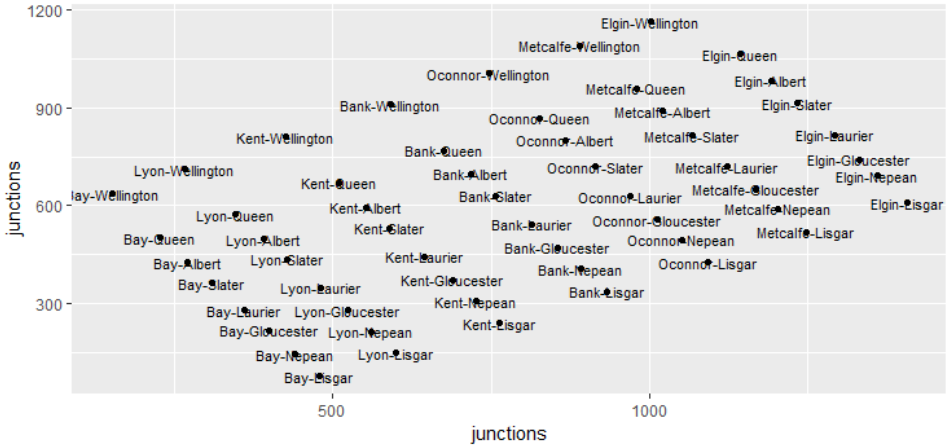

Figure 8 Non-clustered intersections

In figure 9, the network is shown after clustering process has been performed using complete link algorithm and intersections which belong to the same cluster have the same symbol for demonstration. We can see that, the complete-link algorithm produces tightly bound or compact network (1-2) junctions per cluster, from a pragmatic viewpoint, it was observed that, the complete link algorithms are more useful in producing hierarchies in many applications [14].

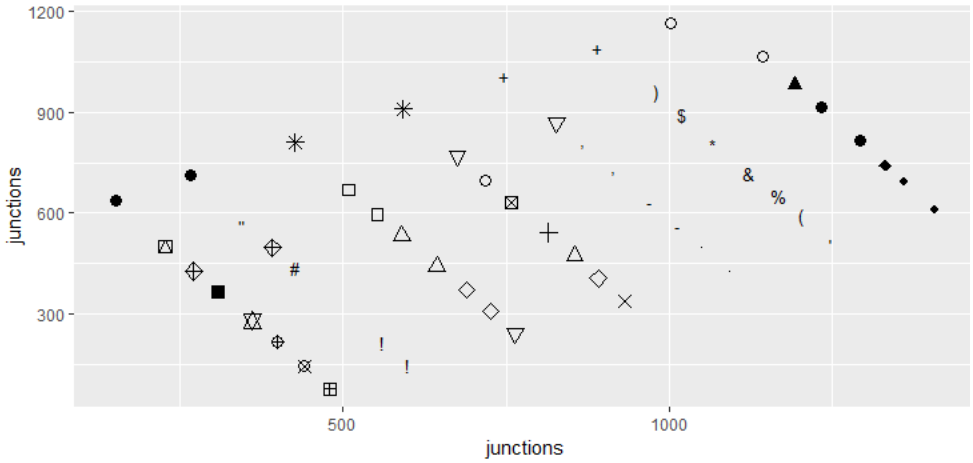

Figure 9 Clustered intersections using complete linkage

After performing the clustering, the network training phase was put into action. In order to do that we used Keras framework [15] to build the network model, shown in figure 10 is an example where state space is 12 and expected output is 3 actions.

The model was run on Dell G5, 8th Gen Intel® Core™ i7-8750H Quad Core Processor, NVIDIA® GeForce® GTX 1060 with NVIDIA® Max-Q Design Technology, 6 GB GDDR5 video memory and 16 Giga of RAM

It takes as input, the state space of the controlled junctions and outputs one of the actions, in this case, the phase to be applied at each junction.

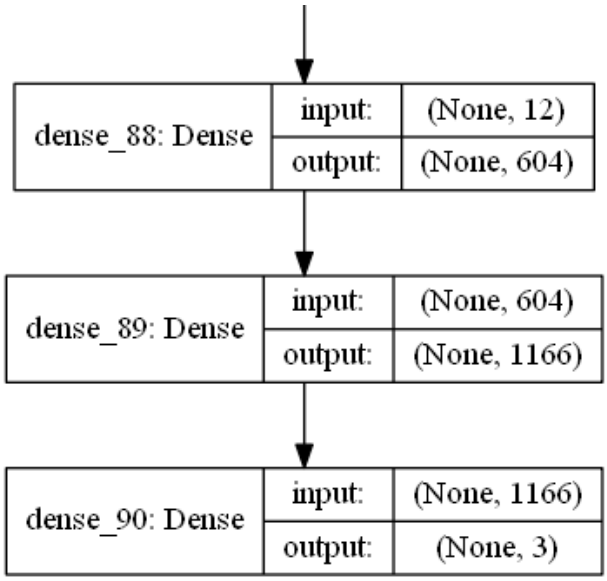

Figure 10 DQN

Clustering phase resulted into having a total of 40 agent controlling the 57 junction, having each agent controlling from 1 junction to max of 2 junctions.

In order to monitor the training process we used Tensorboard [16], in figure 11 we show a sample of training process of 6 agents where X-axis shows the epoch duration and Y-axis is the loss.

Figure 11 agents training process

While using global function approximations for Q-values can speed up learning by generalization, the guarantees of original convergence of Q-learning will no longer hold because divergence and/or oscillation may occur due to going from tabular Q-learning to function approximation, as the model shifts from a tabular representation - one where each (s,a) pair has a local entry – to a global representation, where each (s,a) -pair is evaluated by an approximator that is updated globally. Since in function approximation the weights are updated globally, earlier progress on (s,a) pair can be reverted by updating after sampling another (s',a') pair.

Figures 12, 13 illustrate the throughput and mean travelling time of the clustered-DRL controlled network vs. optimized, static and non-clustered networks respectively.

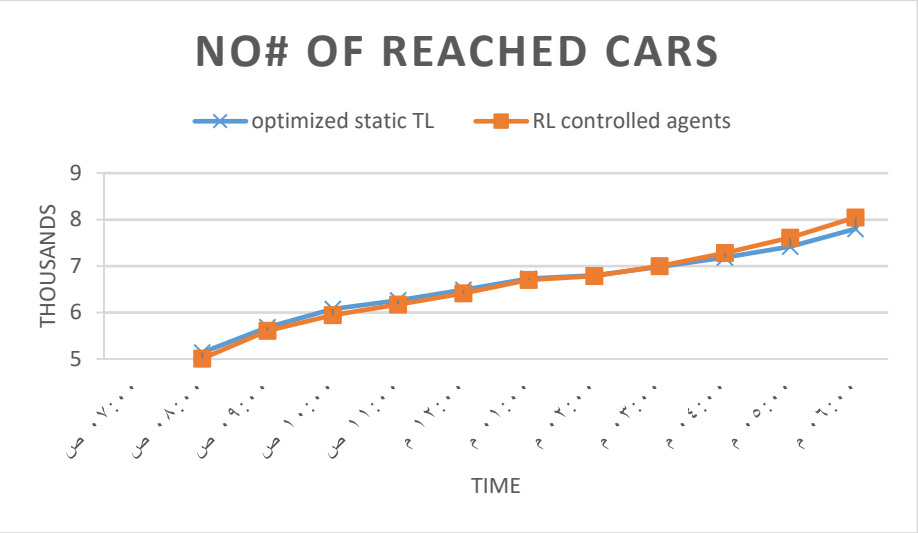

Figure 12 Throughput rate over day

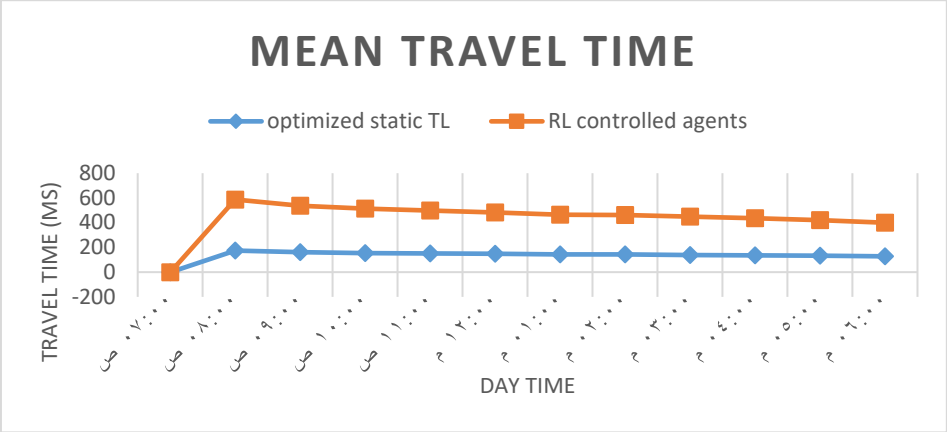

Figure 13 Mean travel time over day

DRL controlled clustered network seems to achieve higher throughput with about ~5% increase than static controlled network and in figure 13 we can see the mean travelling time showing that the original network provided less average travelling time. This happens because green signal stays longer (green wave) [17] in the static network, which results in reducing the average travelling time across the network.

7. Conclusion and future work

Generally, it is not easy to define appropriate state-action spaces in all real-world RL problems. Usually the tiling of the state space has to be rather fine to cover all possibly relevant situations and there can also be a wide variety of actions to choose from. Therefore, there exists a combinatorial explosion problem when trying to explore all possible actions and states.

In our approach to improve the traffic signal controllers system and to overcome reinforcement learning challenges which are caused mainly due to exponential growth in the state-action space resulting in more communications between agents and increased learning time, the network were clustered using Hierarchal clustering algorithm instead of using traditional techniques which are based on geographical attributes, into

strongly connected sub-networks using the traffic volume as a similarity measure, we then used real data to build our simulation scenario and acquire results.

A clustered network controlled by Deep Reinforcement Learning agents was compared against the same network controlled by static and optimized Traffic Light controllers (TLS), the DRL controlled network had an increase of departed vehicles (reached its destination) with a value of 5%.

The suggested technique resulted in higher network throughput but more travelling time. Yet, following the clustering mechanism helped decreasing the number of state-space actions dramatically from $3.7e^{24}$ in the case of single agent controlling all intersections to 3-36 action after clustering and having multiple agents instead. This shall decrease hardware requirements like memory and decrease required learning time during training phase. Another direction for future work is to try different reward functions where we can consider other parameters like traveling time and tune DNN parameters in order to achieve higher throughput and lower travelling time.

References

[1] Kuyer, Lior, Shimon Whiteson, Bram Bakker, and Nikos Vlassis. "Multiagent reinforcement learning for urban traffic control using coordination graphs." In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, PP. 656-671, Year 2008.

[2] Labadie, John W. "Advances in Water Resources Systems Engineering In Applications of Machine Learning." Modern Water Resources Engineering. Humana Press, PP. 467-523. Year 2014

[3] El-Tantawy, Samah, Baher Abdulhai, and Hossam Abdelgawad. "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): methodology and large-scale application on downtown toronto." In: Intelligent Transportation Systems, IEEE Transactions Vol.14 No.3, PP. 1140-1150, Year 2013.

[4] Behrisch, Michael, et al. "SUMO–Simulation of Urban MObility." The Third International Conference on Advances in System Simulation (SIMUL 2011), Barcelona, Spain. 2011.

[5] Carbonell, Jaime G., Ryszard S. Michalski, and Tom M. Mitchell. "An overview of machine learning." In Machine learning, pp. 3-23. Morgan Kaufmann, Year, 1983.

[6] Sutton, Richard S., and Andrew G. Barto. Introduction to reinforcement learning. Vol. 2, No. 4. Cambridge: MIT press, 1998.

[7] Watkins, Christopher JCH, and Peter Dayan. "Q-learning." Machine learning 8, no. 3-4 (1992): 279-292.

[8] Messer, Carroll J., and Ramanan V. Nageswara. Improved traffic signal coordination strategies for actuated control. No. SWUTC/96/465110-1. In Southwest Region University Transportation Center, Center for Transportation Research, University of Texas, Year, 1996.

[9] Mousavi, Seyed Sajad, Michael Schukat, and Enda Howley. "Traffic light control using deep policy-gradient and value-function-based reinforcement learning." IET Intelligent Transport Systems Vol.11, No. 7 PP.417-423, Year 2017.

[10] Lin, Yilun, Xingyuan Dai, Li Li, and Fei-Yue Wang. "An efficient deep reinforcement learning model for urban traffic control." arXiv preprint arXiv: 1808.01876, Year 2018.

[11] Van der Pol, Elise, and Frans A. Oliehoek. "Coordinated deep reinforcement learners for traffic light control." Proceedings of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016) Year 2016.

[12] Genders, Wade, and Saiedeh Razavi. "Using a deep reinforcement learning agent for traffic signal control." arXiv preprint arXiv: 1611.01142 Year 2016.

[13] F.Taher et al Comparing different techniques for controlling traffic signals In International Journal on Power Engineering and Energy (IJPEE), Vol.7, No.3, PP. Year 2016

[14] Jain, Anil K., and Richard C. Dubes. Algorithms for clustering data. Vol. 6. PP. Englewood Cliffs, NJ: Prentice hall, 1988.

[15] Chollet F. keras. GitHub repository. https://github. com/fchollet/keras>. Accessed on. 2015;25:2017.

[16] Chen, Hong-Yunn. "TensorFlow–A system for large-scale machine learning." In OSDI, Vol. 16, PP. 265-283, Year. 2016.

[17] Wu, Xiaoping, Shuai Deng, Xiaohong Du, and Jing Ma. "Green-wave traffic theory optimization and analysis. In "World Journal of Engineering and Technology Vol.2, No. 03 Year 2014.

تطبيق التقسيم الهرمي علي اشارات المرور المحكومة بنظم التعليم المعزز

م./ فادي أحمد – أ.د./ أيمن السيد– د./ أحمد المحلاوي– د./ أحمد شومان

قسم هندسة وعلوم الحاسبات – كلية الهندسة الإلكترونية – جامعة المنوفية.

الازدحام المروري مشكلة خطيرة تؤثر على المدن في جميع أنحاء العالم. حيث انها تزداد سوءا بسبب التزايد المستمر وبشكل كبير في عدد السكان وعدد المركبات. حاليا تعتبر وحدات التحكم (اشارات المرور) هي الآلية الأكثر أهمية للسيطرة على حركة المرور، وتحديدا عند التقاطعات المرورية.

مجال تعليم الالةيوفر المزيد من الأساليب المتقدمة التي يمكن تطبيقها لتوفير المزيد من المرونة وجعل وحدات التحكم أكثر تكيفا لحالة المرور. ويمكن تدريب وحدات التحكم (إشارة المرور) على التكيف للحالة المرورية بطريقة فعالة باستخدام نهج تعزيز التعلم والوكلاء المتعددين و الذي يتعامل مع كل وحدة تحكم علي انها وكيلا وكل وكيل مسؤول عن السيطرة على إشارات المرور حول تقاطع معين. المشكلة العامة لنهج تعزيز التعلم هي الحاجة إلى التنسيق بين الوكلاء والنمو الهائل في كمية الاحتمالات التي يمكن للوكيل ان ينفذها. ويبين هذا البحث أن تقسيم الشبكة المرورية المستهدفة إلى شبكات أصغر باستخدام التقسيم الهرمي يمكن أن يساعد في حل مشكلة كمية الاحتمالات الهائلة وتحسين الأداء الكلي للشبكة