Decentralized Multi Agent Deep Reinforcement Q-learning for Intelligent Traffic Controller

Thamilselvam B*, Subrahmanyam Kalyanasundaram**, and M.V. Panduranga Rao

Dept of Computer Science and Engineering, IIT Hyderabad, India, 502285
`{cs17resch11005,subruk,mvp}@iith.ac.in`

Abstract. Recent development of deep reinforcement learning models has impacted many fields, especially decision based control systems. Urban traffic signal control minimizes traffic congestion as well as overall traffic delay. In this work, we use a decentralized multi-agent reinforcement learning model represented by a novel state and reward function. In comparison to other single agent models reported in literature, this approach uses minimal data collection to control the traffic lights. Our model is assessed using traffic data that has been synthetically generated. Additionally, we compare the outcomes to those of existing models and employ the Monaco SUMO Traffic (MoST) Scenario to examine real-time traffic data.
Finally, we use statistical model checking (specifically, the MultiVeStA) to check performance properties. Our model works well in all synthetic generated data and real time data.

Keywords: Multi agent systems · Deep Reinforcement learning · Statistical Model checking · Traffic controller.

1 Introduction

Traffic management is one of the most important and challenging tasks in urban areas. Efficient operation of traffic lights is crucial in reducing congestion, improving road safety, and enhancing the overall mobility of the city. Conventional traffic light control methods, such as fixed-time [8], [14] and actuated control [7], [13], [5], have been widely used for many years. However, these methods suffer from several limitations, including suboptimal performance, inflexibility, and sensitivity to unexpected events. Traditional centralized approaches to traffic light control have several limitations. For example, large grid sizes hamper the effectiveness of the centralized control strategy they often result in sub-optimal solutions due to the difficulty of modeling complex traffic dynamics.

* Thamilselvam is supported by the DST NM-ICPS, Technology Innovation Hub on Autonomous Navigation and Data Acquisition Systems: TiHAN Foundation at Indian Institute of Technology IIT Hyderabad.

** Subrahmanyam is supported by DST-SERB through the projects MTR/2020/000497 and CRG/2022/009400.

To address these limitations, decentralized multi-agent reinforcement learning (RL) has emerged as a promising approach for traffic light control [20]. The RL framework consists of multiple traffic signals acting as independent agents based on the information shared by its neighbours, each of which is responsible for controlling the timing of its own traffic lights. Moreover, multi-agent RL enables the traffic lights to learn to coordinate their actions, taking into account the impact of their decisions on the flow of vehicles in their vicinity and the network as a whole. These decentralized approaches allow for local decision-making and adaptation to changing traffic conditions, leading to improved traffic flow and reduced congestion.

Despite its potential, the application of decentralized multi-agent RL to traffic light control is still in its infancy, with limited work on the subject to date. For example, the linear weighted function [18] used to optimize an agent's reward may not fully capture the nonlinear throughput relationship between neighboring intersections. This limitation could potentially lead to biased optimal solutions. Moreover, the properties of the trained agent may not be satisfied. For instance, it is challenging to ensure that the maximum waiting time of vehicles does not exceed a certain threshold value.

In this paper, we propose employing Statistical Model Checking (SMC) [9] techniques in conjunction with RL for better traffic policies. The gap that RL leaves in terms of checking properties (like fairness) of a traffic policy is filled by SMC. These properties are established through SMC and the output is fed to the RL reward function. We present a comprehensive and decentralized multi-agent RL framework integrated with MultiVeStA [12] for traffic light control tested in both with synthetic and real world traffic scenario. The other problem we address in this paper is of large state representation in the RL model. Not only are such models computationally expensive, they require a lot of data as well. We develop models that use smaller state representation and reward functions that use only traffic delays only as in input. We experiment with our models and show that they compare favorably against the traditional single agent RL, actuated and fixed traffic light controllers. The idea of using RL coupled with SMC can be generalized to other domains through checking properties of models and including the results in the feedback.

The remainder of this paper is organized as follows: Section 2 briefly discusses some prerequisites for the paper. Section 3 reviews related work in the field of decentralized multi-agent RL for traffic light control. Section 4 presents the proposed decentralized multi-agent RL framework with MultiVeStA. Section 5 describes the simulation environment and experimental setup and Results and analysis. Section 6 concludes the paper with a discussion of future directions.

2 Preliminaries

In this section, we provide an overview of the preliminary concepts and mathematical frameworks required to understand the proposed decentralized multi-agent Reinforcement Learning (RL) framework for traffic light control.

2.1 Machine Learning Background

Reinforcement Learning: RL is a machine learning framework for learning from interaction with an environment [15] over a sequence of time steps, where at each time step t, the agent observes the state s_t of the environment and selects an action a_t based on its policy π. The policy π is a function mapping the state to action. The action a_t then influences the state of the environment, resulting in a new state s_{t+1} and a reward r_{t+1}. The goal of the agent is to learn a policy π that maximizes the expected cumulative reward, defined as:

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \right] = \mathbb{E}_\pi \left[r_1 + \gamma r_2 + \gamma^2 r_3 + \ldots \right] \tag{1}$$

where $\gamma \in [0, 1]$ is the discount factor, which determines the importance of future rewards. The expectation is taken over the distribution of states and actions induced by the policy π. Markov Decision Processes (MDPs) provide a mathematical framework for modeling [1].

Q-Learning: Q-learning [19] is a popular reinforcement learning (RL) algorithm that is used for solving problems in which an agent must learn to make decisions in an environment. In Q-learning, the agent learns a Q-function, which calculates the expected reward for taking an action in a given state.

The Q-function is typically represented as a table or a function that takes the current state and action as inputs and outputs a predicted reward. During training, the agent interacts with the environment, observing the current state, taking an action, receiving a reward, and transitioning to a new state.

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right] \tag{2}$$

Here, $Q(s, a)$ is the estimated value after taking an action a in state s, α is the learning rate, r is the reward received by taking action a in state s, γ is the discount factor, s' is the next state, and a' is the next action. The $\max_{a'} Q(s', a')$ term represents the maximum expected future reward from that next state s'.

Multi-Agent Reinforcement Learning: Multi-Agent RL [16] refers to a scenario where multiple agents interact with each other and with the environment to achieve their goals. In such scenarios, the actions of one agent can affect the state of the environment and the rewards received by other agents.

Decentralized Control: Decentralized control refers to a control system where multiple agents make decisions independently and locally, without relying on a central controller. Decentralized control can improve the scalability and robustness of control systems, as well as reduce the computational burden of centralized controllers.

With these concepts in mind, we now proceed to present the proposed decentralized multi-agent reinforcement learning framework for traffic light control.

2.2 Statistical Model Checking:

Statistical Model Checking (SMC) is a formal methods paradigm that provides a tool for a rapid analysis of properties expected out of (stochastic) systems [9]. The systems can be modeled as some appropriate variant of a transition system, or even a Discrete Event Simulator and the expected properties are expressed as queries written in an appropriate formal logic.

2.3 Tools

SUMO: Simulation of Urban MObility [11] is a microscopic traffic simulator widely used in the field of traffic engineering and transportation research. In addition to its standalone simulator, SUMO also offers a TraCI (Traffic Control Interface) interface, which allows external programs to interact with SUMO simulations in real-time. This is particularly useful for reinforcement learning (RL) applications, where agents need to observe the state of the simulation and take actions in response. The simulation then updates its state based on the agent's action, and the process repeats for the next step.

MultiVeStA: MultiVeStA [12] is a powerful statistical analysis tool that simplifies the integration of automated statistical analysis techniques from the Statistical Model Checking family with existing discrete-event simulators and agent-based models. It supports the use of temporal logic queries, including Probabilistic Computation Tree Logic (PCTL) and Continuous Stochastic Logic (CSL), as well as quantitative temporal expression queries such as Multiquatex. It also supports various analyses, including transient analysis, Counterfactual analysis etc. Integration of SUMO and MultiVeStA is done in [17], which we use in this work.

3 Previous Work

Li et al. [10] proposes a decentralized deep reinforcement learning (RL) approach for optimizing traffic signal timing. The authors present a novel method for using deep neural networks to approximate the Q-values of each traffic light in a decentralized RL framework,where each traffic light is treated as an independent agent that makes decisions based on local observations. This allows the system to operate in a distributed manner, making it suitable for large-scale networks with multiple traffic lights. But deep reinforcement learning algorithms can be sensitive to changes in the environment, and may not perform well in uncertain or dynamic traffic conditions. This can make it difficult to achieve robust and reliable performance in real-world traffic management systems.

Chen et al. [3] demonstrates the scalability and efficiency of the proposed algorithm in handling the complexity and heterogeneity of large-scale urban traffic systems. Although the authors evaluate the performance of the proposed algorithm using a large-scale simulation environment and real-world traffic data, its implementation in real-world traffic systems remains a challenge due to the complexity and heterogeneity of urban traffic systems. But this paper uses a relatively simple state representation that only considers the current traffic flow and the waiting time of vehicles at each intersection. This may not be sufficient to capture the complex interactions and dependencies between traffic signals in large-scale traffic systems and uses a simple reward function that only considers the average waiting time of vehicles at each intersection. This may not be sufficient to capture the complex objectives and trade-offs of traffic signal control, such as balancing the smooth flow of vehicles and reducing congestion and emissions.

Chen et al. [2] proposes a decentralized deep RL approach for traffic signal control, where each traffic signal is treated as an independent agent that learns its optimal control policy through deep RL. The approach is based on deep Q-network (DQN) algorithms. The authors evaluate the performance of the proposed algorithm using a large-scale simulation environment with real-world traffic data and compare the results with conventional methods such as fixed-time and actuated control. The proposed algorithm was evaluated using a large-scale simulation environment with real-world traffic data, but there is no real-world implementation of the algorithm reported in the paper. This limits the ability to assess the scalability and robustness of the algorithm in real-world scenarios and the traffic patterns can change over time, and the proposed algorithm does not take into account the potential for non-stationary traffic patterns.

A deep reinforcement Q-learning model [6] is suggested for enhancing traffic flow at an isolated intersection. The model takes into account partial observation of the environment and its outcomes are compared with those of full detection. However, the experiment's results are derived from a simulated traffic scenario from single intersection and do not reflect real-time traffic conditions.

An existing approaches try to maximize the reward of the agent without checking the properties of the agent. Since agent learns through experiences it is important to check the reliability of the decision. Our approach provides a solution for verifying the properties of an agent while it learns by interacting with statistical model checking (SMC). Subsequently, we evaluate the performance of our model by comparing an agent trained through SMC with an agent trained using the standard approach.

4 RL and SMC based Approach

This section provides an overview of our approach. We use MultiVeStA for training and evaluating the RL agent model. In the training phase an RL agent collects data from SUMO and maximizes its expected reward. Additionally, it collects the output of a MultiQuatex query from MultiVeStA. The query typically would concern an important property of the RL agent's current state. In this work, we illustrate this using two queries. For a reward structure that is based on the change in cumulative waiting time between steps (minimal green time or action), the final reward is calculated as a weighted sum of SUMO data and MultiVeStA.

Reward without considering MultiVeStA output is given by:

$$\sum_{i=T-minGT}^{T} \sum_{j=1}^{N} WT_{i,j} - \sum_{i=T}^{T+minGT} \sum_{j=1}^{N} WT_{i,j}$$

where T is the current step, $minGT$ is the minimal Green Time, N is the number of incoming lanes and WT is the waiting time.
We modify the reward function as follows:

$$\sum_{i=T-minGT}^{T} \sum_{j=1}^{N} WT_{i,j} - \sum_{i=T}^{T+minGT} \sum_{j=1}^{N} WT_{i,j} - x \times (\text{output from MultiVesTA})$$

where x is a multiplicative factor that decides the importance of the query.

Finally, the MultiVeStA is again employed in the testing phase to evaluate the solution. Fig 1 illustrates this approach.

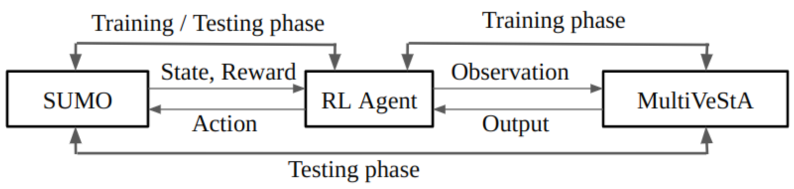

Fig. 1. Interconnection among SUMO, RL Agent and MultiVeStA

4.1 Multi Agent DQN Model for Traffic Signal Controller

Agent actions and Action space: The agent's actions in this context would involve deciding which traffic light phases to activate, based on the current state of the intersection, namely, the number of vehicles and pedestrians waiting at each approach. A phase is defined as a green signal to allow to move vehicles in some directions. For a pictorial illustration of phases, refer to Fig 2. For example, in a four-legged intersection, P1 (phase 1) decides to allow vehicle movement from the north to all other directions (east, south, and west). P5 (phase 5) controls pedestrian movement. When P5 is green, all pedestrians at the intersection are allowed to move in any direction, while all other phases are red.

In this study, we focus on intersection scenarios with three and four legs. Specifically, for three-legged intersections, we analyze four distinct phases, while for four-legged intersections, we consider five phases. The agent responsible for selecting the subsequent phases uses a minimum green interval (also called a *step*). If the agent selects the ongoing phase, the current green interval is extended by the minimum green interval. On the other hand, if a different phase is chosen, the agent switches to the next phase with an intermediate yellow phase that has a minimal yellow interval. Consequently, the agent not only determines the next phase but also controls the duration of the ongoing phase by repeatedly selecting the same phase.

The action space is defined as the set of possible phases in a traffic program. As mentioned earlier in Fig 2, a three-legged intersection has four possible actions, while a four-legged intersection has five possible actions. A phase is controlled by an action.

For a three-legged intersection, the action space is

$A = [(\text{W} \rightarrow \text{EN}), (\text{E} \rightarrow \text{WN}), (\text{N} \rightarrow \text{EW}), (\text{all pedestrian crossing})]$. Note that $|A| = 4$.

For a four-legged intersection, the action space is

$A = [(\text{N} \rightarrow \text{SEW}), (\text{E} \rightarrow \text{WNS}), (\text{S} \rightarrow \text{NWE}), (\text{W} \rightarrow \text{ENS}), (\text{all pedestrian crossing})]$, with $|A| = 5$.

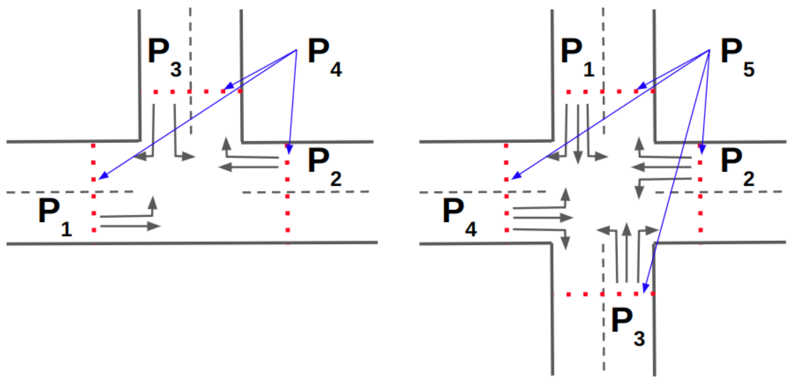

Fig. 2. An intersection with three/four legs with corresponding phases

State representation: We encode the state representation of an intersection's traffic light controller as an array of numbers, which encompasses various variables. These variables consist of the number of waiting vehicles and pedestrians approaching the intersection, the number of vehicles and pedestrians arriving at the intersection. Two observable categories exist, one within a 100-meter radius and the other within a 400-meter radius of the intersection. The state is array of size 20 representing waiting and arriving vehicle's list at an intersection.

Reward function: We use the reward function as described in the beginning of the section, taking SUMO data and MultiVeStA output as input parameters.

Neural Network Architecture: In this approach, the neural network is trained on a dataset of traffic flow information, as per the state representation discussed previously. The neural network is designed to learn patterns in this data and predict the optimal timing of traffic lights for each intersection. By using a neural network to optimize the timing of traffic lights, it is possible to reduce delays and congestion at intersections, as well as improve the flow of traffic. However, it is important to note that the effectiveness of this approach depends on the quality of the data used to train the neural network, as well as the architecture and parameters of the neural network itself. We use the following parameters for our architecture. In the simplest setting, we train a single agent on a single intersection and test the model on multiple intersections. We call this setting the Single-RL. For such a scenario, the neural network architecture comprises an input size of 20 (state representation), and an output size of 4 (actions) for three-legged intersections and 5 (actions) for four-legged intersections. The network comprises 4 hidden layers, each containing 400 neurons, with ReLU activation function. The optimizer used is Adam, with a learning rate of 0.001, and a batch size of 100. The discount factor used for future rewards is 0.75. The replay buffer size is set to 50000, allowing the network to learn from previous experiences. These hyperparameters are crucial in determining the performance of the neural network in the reinforcement learning task, and they can be fine-tuned to improve the learning efficiency and the overall performance of the network.

4.2 Decentralized Multi Agent Model

In decentralized Multi Agent systems agents can transfer their local data to neighboring agents. These agents then make decisions based on all the available data. We classify the agents into two categories: bordered and non-bordered. Bordered agents can receive data from only one neighbor, while non-bordered agents can obtain data from two neighbors located on the major or arterial road.

In our approach, we include additional data in the reward function of the model, as well as in the state representation. The state of the model consists of information on the current phase of neighboring agents, its geographical location, and the time elapsed during that phase. This state representation is adequate for independent model training and decision-making without requiring a centralized system. The reward includes delays from other intersections as well.

4.3 MultiVeStA with Reinforcement Learning

Simple Query: What is the probability that the green time of a particular phase has crossed the threshold when there are vehicles waiting in the other direction at an intersection? This query is useful to check the fairness/liveness property of the RL agent. Since the RL agent tries to maximize the reward, if there are very less number of vehicles waiting in one direction and more number of vehicles waiting in another direction, the RL agent chooses to extend the current phase for heavy traffic side. This makes the side with less traffic to wait

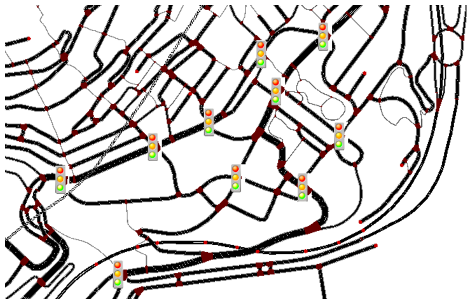

Fig. 3. Monaco SUMO Traffic (MoST) Scenario

more, which is undesirable. Result of the query is shown in Table 2, from the table we can see that the model trained with MultiVeStA yields less probability than the one without MultiVeStA, in both Low and High traffic scenarios. But at the same time it compromises the average waiting time. The ratio of the query satisfied between model trained with and model not trained with MultiVeStA is more in High than Low traffic scenario because normal RL agent tries to reduce the delay:

```
EMG(st) = if ( s.rval("step") <= st ) then
             ( if (s.rval("gtExceed") == 1 then (1)
               else (0) fi)
          else #EMG((st)) fi ;
eval E[ EMG(st) ] ;
```

Listing 1.1. Probability of $MAX - GREEN$ crossed in particular phase.

```
t1Ut2(st,th,cp) = if( s.rval("step") <= st) then
        if ( s.rval("diff") > th ) then (1)
        else if ( s.rval("curPhase") == cp ) then
                 #t1Ut2((st),(th),(cp))
             else  (0) fi fi else (0) fi ;
eval E[t1Ut2(st,th,cp)];
```

Listing 1.2. Probability query using the Until Operator

Complex Query

What is the probability that the current phase will be prolonged, provided that the absolute difference between the number of vehicles waiting in the current phase and the other phases is a specified threshold (say 5)? This query assesses the potential for grouping vehicles in a single direction, enabling drivers approaching the traffic signal to make informed decisions about whether to slow down or speed up.

5 Experimental Results

In our experiments, we consider the two kind of traffic scenarios, one is synthetic and another one is real time data from Monaco city [4]. For synthetic generated traffic data, we use a six intersections grid road network with length of 550m. For training the model, we use Weibul distribution for vehicle generation of 1000 (500) per hour for High (Low). We compare the result of all the algorithms in terms of cumulative Waiting Time (WT) in seconds, averaged over the six intersections and cumulative Queue Length (QL).

5.1 Decentralized Multi-Agent Model

In this experiment, we evaluate our reward function structure for the decentralized RL formulation. Table 1 shows the results.

Decentralized RL performs well, because it learns relationships among neighbours and takes action which leads to overall minimisation of delay and queue length.

The results for Monaco city are shown in Fig 4. The cumulative delay at intersection level varies because the decentralized RL controller aims to reduce overall waiting time of all intersections. This can be seen through the mean waiting time (the dashed line of the controller).

Table 1. Waiting time and Queue length of all controllers in Grid Road Network

	Fixed-Time		**Actuated**		**Single-RL**		**Decentralized-RL**	
	WT(s)	**QL**	**WT(s)**	**QL**	**WT(s)**	**QL**	**WT(s)**	**QL**
Low	8498	48314	6375	34343	4206	23810	3728	19678
High	20887	134988	14404	91790	10273	64878	8340	58965

5.2 Reinforcement Learning With MultiVeStA

We use only one intersection in this case, to focus on the effect of employing inputs from MultiVeStA to the RL agent. Table 2 shows the results. *Prob* indicates the probability that the corresponding MultiQuatex query is satisfied. The multiplicative factor x used is -1 for the first query and $+1$ for the second query. Considering the queries, the signs are a natural choice; we choose the magnitude after some trials. The results indicate an advantage of incorporating Statistical Model Checking tools into RL systems.

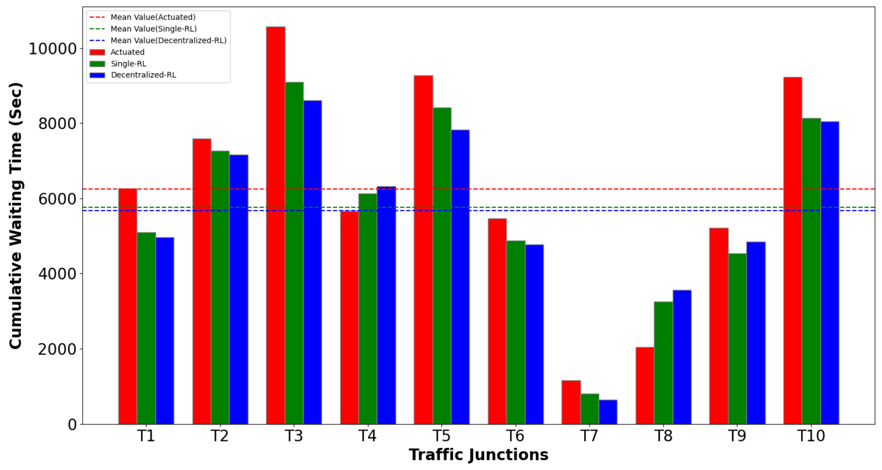

Fig. 4. Average Waiting Time at Intersections in the MOST (Monaco city) Scenario

Table 2. Result of Evaluation of Queries

	Model Trained Without MultiVeStA				**Model Trained With MultiVeStA**			
	Query-1		**Query-2**		**Query-1**		**Query-2**	
	Prob	**WT(s)**	**Prob**	**WT(s)**	**Prob**	**WT(s)**	**Prob**	**WT(s)**
Low	0.06	1.042	0.68	1.054	0.03	1.398	0.72	1.042
High	0.025	2.258	0.76	1.364	0.008	2.481	0.88	1.174

6 Future Directions

Our results indicate an encouraging new direction for using decentralized multi agent systems in conjunction with statistical model checking tools for traffic analysis. We believe that the following questions are interesting directions to pursue. What properties (MultiQuatex queries in our case) of the RL agent are best for fast convergence? We arrived at the multiplicative factor for the MultiVeStA input through trial and error. Is there a systematic approach to converge to the best value? How does our technique scale to large cities?

References

1. Bellman, R.: A markovian decision process. Journal of mathematics and mechanics pp. 679–684 (1957)
2. Chen, C., Wei, H., Xu, N., Zheng, G., Yang, M., Xiong, Y., Xu, K., Li, Z.: Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 3414–3421 (2020)

3. Chen, Y., Li, C., Yue, W., Zhang, H., Mao, G.: Engineering a large-scale traffic signal control: A multi-agent reinforcement learning approach. In: IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). pp. 1–6 (2021)
4. Codeca, L., Härri, J.: Monaco SUMO Traffic (MoST) Scenario: A 3D Mobility Scenario for Cooperative ITS. In: SUMO 2018, SUMO User Conference, Simulating Autonomous and Intermodal Transport Systems, May 14-16, 2018, Berlin, Germany. Berlin, GERMANY (05 2018)
5. De Schutter, B.: Optimizing acyclic traffic signal switching sequences through an extended linear complementarity problem formulation. European Journal of Operational Research **139**(2), 400–415 (2002)
6. Ducrocq, R., Farhi, N.: Deep reinforcement q-learning for intelligent traffic signal control with partial detection. International Journal of Intelligent Transportation Systems Research pp. 1–15 (2023)
7. Gallivan, S., Heydecker, B.: Optimising the control performance of traffic signals at a single junction. Transportation Research Part B: Methodological **22**(5), 357–370 (1988)
8. Gazis, D.C.: Optimum control of a system of oversaturated intersections. Operations Research **12**(6), 815–831 (1964)
9. Legay, A., Lukina, A., Traonouez, L.M., Yang, J., Smolka, S.A., Grosu, R.: Statistical Model Checking, pp. 478–504. Springer International Publishing (2019)
10. Li, Z., Xu, C., Zhang, G.: A deep reinforcement learning approach for traffic signal control optimization. arXiv preprint arXiv:2107.06115 (2021)
11. Lopez, P.A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., Wießner, E.: Microscopic traffic simulation using sumo. In: The 21st IEEE International Conference on Intelligent Transportation Systems. IEEE (November 2018)
12. Sebastio, S., Vandin, A.: Multivesta: statistical model checking for discrete event simulators. In: Horváth, A., Buchholz, P., Cortellessa, V., Muscariello, L., Squillante, M.S. (eds.) 7th International Conference on Performance Evaluation Methodologies and Tools, ValueTools '13. pp. 310–315. ICST/ACM (2013)
13. Sen, S., Head, K.L.: Controlled optimization of phases at an intersection. Transportation science **31**(1), 5–17 (1997)
14. Smith, M.: Traffic control and route-choice; a simple example. Transportation Research Part B: Methodological **13**(4), 289–294 (1979)
15. Sutton, R., Barto, A.: Reinforcement learning: An introduction. IEEE Transactions on Neural Networks **9**(5), 1054–1054 (1998)
16. Tan, M.: Multi-agent reinforcement learning: Independent vs. cooperative agents. In: Proceedings of the tenth international conference on machine learning. pp. 330–337 (1993)
17. Thamilselvam, B., Kalyanasundaram, S., Parmar, S., Panduranga Rao, M.: Statistical model checking for traffic models. In: Formal Methods: Foundations and Applications: 24th Brazilian Symposium, SBMF 2021, Virtual Event, December 6–10, 2021, Proceedings 24. pp. 17–33. Springer (2021)
18. Wang, X., Ke, L., Qiao, Z., Chai, X.: Large-scale traffic signal control using a novel multiagent reinforcement learning. IEEE transactions on cybernetics **51**(1), 174–187 (2020)
19. Watkins, C.J., Dayan, P.: Q-learning. Machine learning **8**, 279–292 (1992)
20. Zhou, P., Chen, X., Liu, Z., Braud, T., Hui, P., Kangasharju, J.: Drle: Decentralized reinforcement learning at the edge for traffic light control in the iov. IEEE Transactions on Intelligent Transportation Systems **22**(4), 2262–2273 (2020)