A Novel Multi-Agent Deep RL Approach for Traffic Signal Control

1st Wang Shijie

School of Advanced Technology Xi'an Jiaotong-Liverpool University Suzhou, China shijie.wang18@alumni.xjtlu.edu.cn 2nd Wang Shangbo* School of Advanced Technology Xi'an Jiaotong-Liverpool University Suzhou, China shangbo.wang@xjtlu.edu.cn

Abstract—As travel demand increases and urban traffic condition becomes more complicated, applying multi-agent deep reinforcement learning (MARL) to traffic signal control becomes one of the hot topics. The rise of Reinforcement Learning (RL) has opened up opportunities for solving Adaptive Traffic Signal Control (ATSC) in complex urban traffic networks, and deep neural networks have further enhanced their ability to handle complex data. Traditional research in traffic signal control is based on the centralized Reinforcement Learning technique. However, in a large-scale road network, centralized RL is infeasible because of an exponential growth of joint state-action space. In this paper, we propose a Friend-Deep Q-network (Friend-DQN) approach for multiple traffic signal control in urban networks, which is based on an agent-cooperation scheme. In particular, the cooperation between multiple agents can reduce the state-action space and thus speed up the convergence. We use SUMO (Simulation of Urban Transport) platform to evaluate the performance of Friend-DQN model, and show its feasibility and superiority over other existing methods.

Index Terms—Traffic Signal Control; Machine Learning; Deep Reinforcement Learning; Decentralized Multi-Agent

I. INTRODUCTION

The rapid development of urbanization has facilitated people's daily travel. Still, at the same time, traffic problems have become more and more serious, and traditional traffic management models and traffic systems are no longer able to meet the actual requirements of the times [1]. In the new context, urban traffic should change in the direction of intelligence, actively introduce advanced artificial intelligence technology, and carry out targeted solutions to the current problems to effectively solve a series of traffic problems.

Traffic signals in urban road networks are almost always fixed-phase and cannot adapt to different traffic conditions, and thus cause congestion at intersections [2]. To relieve urban congestion problems, some literature has applied adaptive traffic signal control (ATSC) strategy to minimize the average waiting time of the urban network by dynamically adjusting signal timing according to real-time traffic state [3], [4]. However, it becomes challenging to dynamically predict the traffic flow and adjust the signals when dealing with massive traffic. Reinforcement learning technique has shown many significant achievements and traffic signal control and management in complex traffic environment [5]–[8]. However, traditional reinforcement learning methods like Deep Q-network (DQN) and Q-learning can become very large in the action space and state space when dealing with complex traffic networks, and in many cases are slow to converge. Therefore, the centralized RL has mainly two drawbacks. The first one is high latency caused by collecting all the traffic measurements in the network and feeding them back to the center for centralized processing. The second one is large space occupation caused by the joint action of the agents as the number of traffic junctions grows [9].

To overcome the limitations, multi-agent RL technique can be applied to urban networks by considering each road intersection as a local RL agent. Although different techniques and algorithms are used for different scenarios like traffic signal control and vehicle signal coordination control, most introduce neural networks in reinforcement learning, using the robust representational power of neural networks to build models [10], [11]. According to Matthew E.Taylor's survey, transportation problems can be defined as the work of Learning cooperation [12], where each agent aims to learn a dominant strategy which is trying to maximize the value function obtained by the traffic network. At the same time, each agent will develop its own strategy with consideration of neighboring agents' strategies. Agents need to learn collaboratively to find a policy maximizing the global reward, instead of maximizing an agent's own reward to reduce the average wait time for all vehicles in the system.

To realize the target, cooperation learning strategy should be applied to multiple intersection signal control problems, that is, learning how to cooperate under incomplete communication conditions. To solve ATSC effectively, we have developed an adaptive intelligent traffic control algorithm using multiagent RL based on improved Friend Deep Neural Network Qlearning, namely, Friend-DQN [13]. The Friend-DQN method does not increase joint state-action space exponentially as the number of intersections increases, and thus, Friend-DQN will converge faster than traditional Q-learning and DQN. More specifically, the main contributions of this paper are:

- We propose a Friend-DQN model to deal with multiintersection problems, which aims to minimize average vehicle waiting time;
- We compare the Friend-DQN model with fixed-phase, centralized DQN and independent DQN for different numbers of intersections in terms of convergence speed

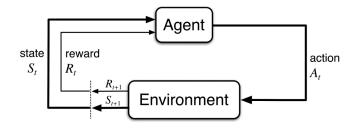


Fig. 1. Interaction of agents with their environment in Markovian decision processes.

and average waiting time;

• We justify the effectiveness and superiority of the Friend-DQN model on the SUMO platform.

II. RELATED WORK

The first academic application of RL technique in a traffic signal control problem was the successful application of SARSA to traffic signal control [5]–[7]. SARSA is an on-policy algorithm for learning a Markov decision process policy, which combines timely and intelligent traffic control policies with real-time road traffic [14]. Srinivasan et al. [15] uses a distributed multi-agent model to solve the traffic signal control problem, where each agent has an independent Q table to learn and judge the execution phase. Experiments demonstrate the effectiveness of Q learning.

Recently, IntelliLight was proposed to be implemented using DQN and tested in a real road network [8]. IntelliLight is combined with a specific traffic signal control problem. The environment consists of traffic signal phases and traffic conditions, and the state is a characteristic representation of the environment information. The agent inputs the state and controls the signals as an action, such as changing the traffic signal phase or the duration of the signal, and then the agent gets a reward from the environment. The agent in IntelliLight implements this through a DQN network, which updates the model based on the loss function of the DQN network to maximize the reward. It is worth noting that the research argues that the agent has to analyze and understand the strategy in the context of the actual scenario. There is no denying that IntelliLight does perform well in real cities. However, it is still a centralized RL algorithm, which means that it still cannot avoid the vast space and time occupation when dealing with a large-scale network.

A multi-agent deep RL method that combines the DQN algorithm with transfer planning can solve the difficulty of centralized RL [16]. Transfer planning can avoid the problems of previous multi-agent reinforcement learning i.e. space and time occupation and allow for faster and more scalable learning. This study introduces a new reward function to the ATSC problem. It solves the problem of extreme delays previously caused by a single average vehicle waiting time as a reward by combining criteria such as transport penalties and vehicle delays with different weights to calculate a new reward.

TABLE I THE PARAMETERS OF FRIEND-DQN

Parameter	Value
Greedy rate	0.9
Maximize epsilon value	0.9
Learning rate	0.1
Memory size	20000
Batch size	50
Target network weight updating frequency	400

Finally, the control of multi-agent is achieved by transfer planning and max-plus coordination algorithms. This approach reduces the problem of large spaces for single agents to some extent, but it is still precarious and sometimes underperforms due to the use of deeper networks.

Recent research has proposed the use of independent advantage actor-critic (A2C) for traffic signal control instead of Qlearning [9]. Although they expanded the state representation by including observations and fingerprints of neighbouring agents in each agent's state and used a spatial discount factor to adjust the global reward for each agent, they did not consider the higher-order relationships of the agents. Others have used the more robust DDPG instead of the A2C method. However, in the past DDPG-based traffic control frameworks [17], [18] focused only on single intersections and could not be applied to large-scale traffic networks.

III. METHODOLOGY

To implement more robust adaptive traffic signal control, we propose the decentralized Friend-DQN algorithm in the framework of reinforcement learning. Since the theory of Friendlearning is an enhancement of Nash-learning, we firstly briefly review fundamentals of MDP and Nash-Q before introducing Friend-learning.

A. Model

It can be observed from some literature [16], [19]-[21] that traffic signal control problem can be described as a Markov Decision Process (MDP). MDP aim to detect the state of the environment, select actions, and associate goals related to the state of the environment in a simple form. The definition of the MDP contains the state space S, action set A, transition probability P and reward R. The agent reacts to an environmental state $s_t \in S$ by taking a possible action $a_t \in A$. It ends up in state s_{t+1} with some transition probability $p(s_{t+1}|s_t, a_t) \in P$ and receives a reward signal $r(s_t, a_t, s_{t+1}) \in R$ [22]. The process is shown in Figure 1. State S: Figure 2a shows a four-intersection traffic signal control problem, where the matrix represents an image in SUMO to represent the state of the vehicles around the intersections. In figure 2b, referring to the definition as Tobias [23], we use a matrix to represent the vehicle position information in the traffic signal controlled lane as the state. The whole two-

dimensional space shown in figure 2a is split into several

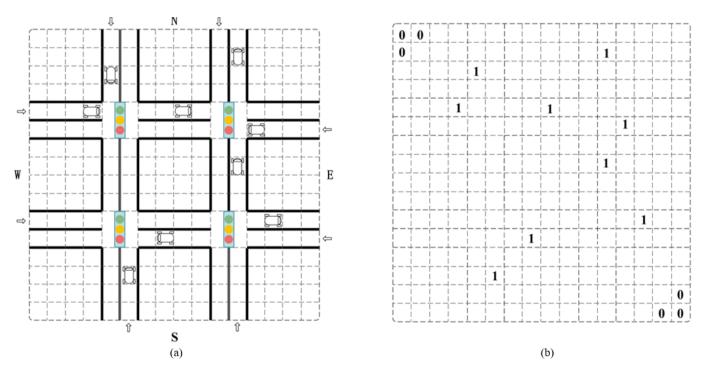


Fig. 2. a) Traffic situation. b) Simplified example of state representation in a 16×16 matrix.

squares with the same length and width, each of which is filled with one or zero representing the existence of vehicles. **Action A**: At each step, agents can choose a traffic signal duration as the action which will change the states. In our design, there are six actions per junction, which are six different phases in five-second intervals from 10 to 30 seconds and can be selected as the phase of the junction traffic signal at each update.

Transition probability P: Transition probability $p(s_{t+1}|s_t, a_t)$ defines the probability of state transition from the current s_t to the next state s_{t+1} when the agent takes action a_t .

Reward R: Reward is defined as the difference the average waiting time of vehicles between the next state and the current state.

The rewards that define the traffic signal control problem are not uniform. Here we take the metric of the reduced waiting time for vehicles at intersections. However, in a real traffic road network, the average waiting time will be calculated when a vehicle finishes its journey. This leads to severe latency problems. So here we select the action at time t and calculate the reward and learning at the next phase, i.e., time t + 1.

The final reward rt for each time step is as follows:

$$r_t = \sum_{n=1}^{N} w_t - \sum_{n=1}^{N} w_{t+1}$$
(1)

where N represents the number of vehicles on the lanes, and w is the waiting time.

The objective of the agent is to maximize the accumulation of rewards. By formalizing the reward, it is passed from the

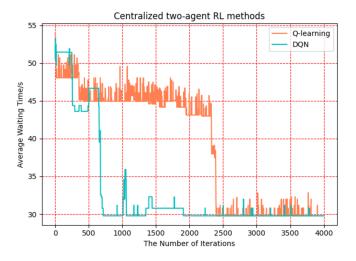


Fig. 3. Centralized two-agent RL method.

environment to the agent. The agent is rewarded with the sum of the rewards:

$$G_t = r_{t+1} + r_{t+2} + r_{t+3} + \dots + r_T$$
(2)

To make the agent more "farsighted", i.e., to consider future rewards, introduce a discount factor gamma, then the agent chooses action A_t at time t to maximize the desired discounted reward:

$$G_t = r_{t+1} + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots = \sum_{0}^{\infty} \gamma^k r_{t+k+1}$$
(3)

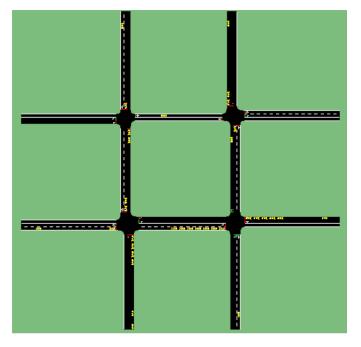


Fig. 4. Traffic network with 4 intersections.

If gamma is equal to 0, then the agent only considers current rewards and the objective of the agent is to learn how to choose an action A_t to maximize r_t .

B. Nash-Q and Friend-Q algorithm

The Multi-agent Nash Q algorithm considers other agents when selecting actions, i.e., it selects the action that makes the system reward and largest. The Nash equilibrium locks in each agent's strategy because it cannot simply change its own strategy to increase its payoffs [24]. The learning agent, indexed by i, learns its Q-value by making arbitrary guesses at moment t. At moment t, agent i takes an action by observing the current state. Afterward, it learns the reward of itself, the actions taken by all other agents, the rewards of others and the new state s'. A Nash equilibrium is then calculated for the current phase and updated the Q-value according to:

$$Q_{t+1}^{i}(s, a^{1}, ..., a^{n}) = (1 - \alpha)Q_{t}^{i}(s, a^{1}, ..., a^{n}) + \alpha_{t}[r_{t}^{1} + \beta NashQ_{t}^{i}(s')]$$
(4)

Where

$$NashQ_{t}^{i}(s') = \pi^{1}(s')...\pi^{n}(s')Q_{t}^{i}(s')$$
(5)

Its updates are asynchronous, that is, only actions relating to the current state are updated.

The convergence condition for the Nash Q-Learning algorithm to converge in a cooperative or adversarial equilibrium setting is that a global optimum or saddle point can be found in each state s of the stage game. The Nash Q-learning algorithm can only converge if this condition is satisfied.

Nevertheless, in a transport network, the relationship between different agents is not just competitive, we want multiple agents to work together to get the most rewards for the whole

TABLE II THE ATTRIBUTES ON SUMO

Attributes	Value
Number of junctions	2/3/4
Number of roads	12
Average length of lanes	100m
Number of lanes per road	3
Phase duration	40s
Arrival distribution	Uniform
Simulation duration	5.5hours

transport system. We, therefore, refer to the Friend or Foe Q-Learning (FFQ) proposed by Littman [25]. Friend-Q assumes that the opponent is like a friend who maximizes everyone's benefits, so add the action space of player B to Q.

$$FriendQ_t^i(s, Q_1, Q_2) = \max_{a_1 \in A_1, a_2 \in A_2} Q[s, a_1, a_2]$$
(6)

By appropriately replacing NashQ with FriendQ, distributed agent systems can achieve a balanced strategy through cooperative learning.

Algorithm 1 Multi-Agent Friend-DQN

- initialization
 Let t = 0, get the initial state s₀.
- 3: Let the learning agent be indexed by k.
- 4: for all $s \in S$ and $a_i \in A_i$, i = 1, ..., n do
- 5: Let $Q_{t}^{i}(s, a^{1}, ..., a^{n}) = 0.$
- 6: end for
- 7: loop
- 8: Choose action a_t^k .
- 9: Observe $r_t^1, ..., r_t^n; a_t^1, ..., a_t^n$, and $s_{t+1} = s'$.
- 10: **for** i = 1, ..., n **do**
- 11: $\begin{aligned} Q_t^i(s, a^1, ..., a^n) &= (1 \alpha) Q_t^1(s, a^1, ..., a^n) + \alpha_t [r_t^i + \beta Friend Q_t^i(s, Q_1, ..., Q_n)] \end{aligned}$
- 12: end for
- 13: Where $\alpha \in (0,1)$ is the learning rate, and $FriendQ_t^i(s, Q_1, ..., Q_n)$ is defined in (8)
- 14: Let t := t + 1

15: end loop

C. Multi-agent Friend-DQN algorithm

However, this is not enough, we also need to extend Friend-DQN so that it can be adapted to larger systems of agents. The Friend-Q value becomes as follows:

$$Friend_{i}(s, Q_{1}, ..., Q_{n}) = \max_{\pi \in \Pi(X_{1} \cdots X_{k})} \sum_{x_{1}, ..., x_{k} \in X_{1} \cdots X_{k}} \pi(x_{1}) ... \pi(x_{k}) Q[s, x_{1}, ..., x_{k}]$$
(7)

FriendQ is a strategy for improving Q-learning to find equilibrium for multi-agent systems. Q-learning needs to generate Q-tables in runtime, so when processing traffic, the large state space can result in the need to generate a colossal q-table. We examined the ATSC of DQN and Q-learning at two junctions

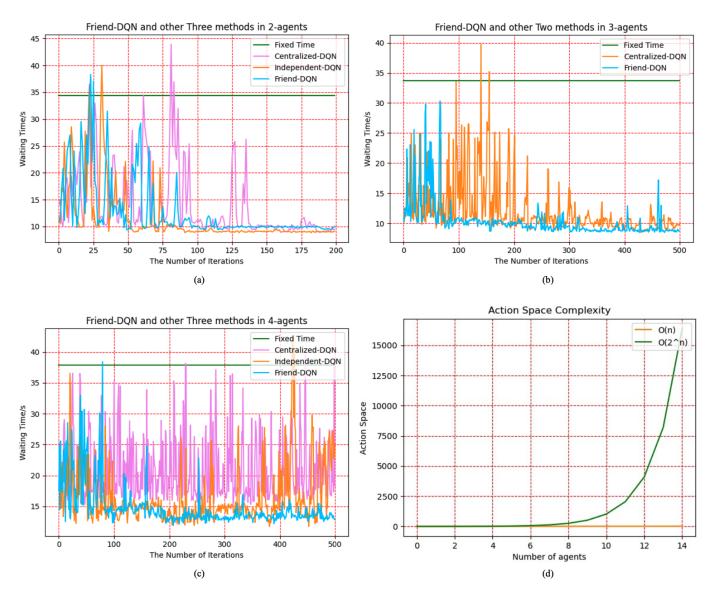


Fig. 5. a) 4 methods in 2-agents. b) 3 methods in 3-agents. c) 4 methods in 4-agents. d) Action Space Complexity for DQN and Friend-DQN

and figure 3 showed that DQN converges much faster than Qlearning due to the neural network it introduces. Therefore, it is necessary to improve Friend-DQN to increase the convergence speed further.

DQN uses neural networks to represent Q values, which is what becomes represented by Q networks. We use the target Q value as a label to get the Q value to converge to the target Q value [26].

Therefore, the loss function for Q-network training is:

$$L(w) = E[(r + \gamma \max_{a'} Q(s', a', w) - Q(s, a, w))^{2}](8)$$

Where $r + \gamma max Q(s', a', w)$ is the target.

We determined the loss function, i.e., cost, and the way to obtain the samples, the whole algorithm of DQN is shaped.

where X represents the set of all agents.

The hyperparameters of Friend-DQN are shown in TableI. Algorithm1 illustrates our proposed algorithm.

IV. EXERIMENTAL RESULTS

Here we choose the SUMO platform for the simulation. Using four traffic lights as an example, see Figure 4, all eight traffic roads will depart at a specific frequency. The system configuration parameters are shown in TableII.

We compared Friend-DQN with a traditional centralized DQN, independent-DQN and a fixed-time ATSC on SUMO. Because the traffic network is asymmetric when there are three agents, the independent method cannot be directly experimented with the traffic network. Thus we compared Friend-DQN with independent-DQN in the two-agents and four-agents settings. The results are shown in Figure 5, which shows that the fixed time is not optimized for traffic. Although the traditional DQN can optimize ATSC, the convergence speed is much slower than the decentralized Friend-DQN algorithm. Moreover, as more junctions are added, the gap between Friend-DQN and centralized DQN grows. At two agents,

the convergence speed of independent-DQN and Friend-DQN is approximated. This is due to that the action spaces of both methods are the same. However, Figure 5(c) shows that the independent-DQN method keeps oscillating unable to converge to a policy when there are four agents. This is because there is no communication between agents and cannot converge to a stable policy.

In addition, Figure 5(c) shows that after 500 trailing epochs at four traffic junctions, the DQN still does not converge. When we analyze the complexity of the action space of these two algorithms, we can see that Friend-DQN outperforms centralized RL to a large extent. Action space represents the projection of the actions in the system. The action space complexity of centralized RL is $O(2^n)$, while the action space complexity of Friend-DQN is only O(n). So when there are four junctions, the centralized approach has 1296 action choices compared to 24 for Friend-DQN. This is why there is such a big difference between the two methods in convergence speed.

Figure 5(d) shows that as the number of traffic junctions increases, the time and space required for centralized algorithms will become a huge hassle. That is why we are developing decentralized multi-agent to implement ATSC.

V. CONCLUSIONS

In this paper, we have demonstrated that Friend-DQN is a promising approach to adaptive traffic signal control. In an entire traffic network, traffic flows are dynamic and change over time. Our proposed approach is based on information in a phase statistic and learning the optimal joint action of multiple agents in different situations. Vehicle location information and queue length are used as collected information. Many states and joint actions are learned in training, so our algorithm can be extended to more extensive traffic networks.

At the same time, the decentralized Friend-DQN algorithm is a scalable multi-agent approach to deep reinforcement learning. Using cooperation to achieve equilibrium avoids the problems of single-agent reinforcement learning and allows for faster and more scalable learning. Its action space complexity is linear, so as more traffic intersections are added, the algorithm performs significantly better than earlier single-agent traffic signal control efforts.

We conducted simulation experiments on SUMO for four junctions and proved the performance and stability of the algorithm. Simulation results also illustrate that our approach outperforms the fixed-time and DQN algorithms in different traffic conditions.

Our current system only considers cooperation between traffic junctions. It can lead to some inequities i. e. excessive waiting times at certain junctions. Future work includes applying the Nash Equilibrium and Friend-DQN so that agents can cooperate and also secure their interests.

ACKNOWLEDGMENT

This work was supported in part by XJTLU Research Development Funding RDF-21-02-015. Any comments should be address to Dr. Wang Shangbo.

REFERENCES

- [1] H. Wei, G. Zheng, V. Gayah, and Z. Li, "A survey on traffic signal control methods," *arXiv preprint arXiv:1904.08117*, 2019.
- [2] C. Li, W. Yue, G. Mao, and Z. Xu, "Congestion propagation based bottleneck identification in urban road networks," *IEEE Transactions on Vehicular Technology*, 2020.
- [3] M. Essa and T. Sayed, "Self-learning adaptive traffic signal control for real-time safety optimization.," Accident Analysis & Prevention, 2020.
- [4] K.-L. A. Yau, J. Qadir, H. L. Khoo, M. H. Ling, and P. Komisarczuk, "A survey on reinforcement learning models and algorithms for traffic signal control," ACM Computing Surveys, 2017.
- [5] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [6] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: a survey," *Journal of Artificial Intelligence Research*, 1996.
- [7] G. A. Rummery, "On-line q-learning using connectionist systems," *CTIT technical reports series*, 1994.
- [8] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: A reinforcement learning approach for intelligent traffic light control," *knowledge discovery and data mining*, 2018.
- [9] T. Chu, J. Wang, L. Codeca, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, 2019.
- [10] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *neural information processing systems*, 2017.
- [11] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," *national conference on artificial intelligence*, 2017.
- [12] P. Hernandez-Leal, B. Kartal, and M. D. Taylor, "A survey and critique of multiagent deep reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, 2018.
- [13] M. L. Littman, "Friend-or-foe q-learning in general-sum games," international conference on machine learning, 2001.
- [14] X. Zhou, F. Zhu, Q. Liu, Y. Fu, and W. Huang, "A sarsa (λ)-based control model for real-time traffic light coordination," *The Scientific World Journal*, vol. 2014, 2014.
- [15] P. Balaji, X. German, and D. Srinivasan, "Urban traffic signal control using reinforcement learning agents," *Iet Intelligent Transport Systems*, 2010.
- [16] E. Van der Pol and F. A. Oliehoek, "Coordinated deep reinforcement learners for traffic light control," *Proceedings of learning, inference and control of multi-agent systems (at NIPS 2016)*, vol. 8, pp. 21–38, 2016.
- [17] H. Pang and W. Gao, "Deep deterministic policy gradient for traffic signal control of single intersection," *chinese control and decision conference*, 2019.
- [18] H. Wu, "Control method of traffic signal lights based on ddpg reinforcement learning," in *Journal of Physics: Conference Series*, vol. 1646, p. 012077, IOP Publishing, 2020.
- [19] R. A. Howard, "Dynamic programming and markov processes.," 1960.
- [20] R. Bellman, "A markovian decision process," Indiana University Mathematics Journal, 1957.
- [21] D. Blackwell, "Discrete dynamic programming," Annals of Mathematical Statistics, 1962.
- [22] J. A. Bagnell, S. M. Kakade, J. Schneider, and A. Y. Ng, "Policy search by dynamic programming," *neural information processing systems*, 2003.
- [23] T. Rijken, DeepLight: Deep reinforcement learning for signalised traffic control. PhD thesis, Master's thesis. University College London, 2015.
- [24] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," arXiv preprint arXiv:1312.5602, 2013.
- [25] P. Coll, P. Factorovich, I. Loiseau, and R. Gómez, "A linear programming approach for adaptive synchronization of traffic signals," *International Transactions in Operational Research*, 2013.
- [26] J. Hu, M. P. Wellman, *et al.*, "Multiagent reinforcement learning: theoretical framework and an algorithm.," in *ICML*, vol. 98, pp. 242– 250, Citeseer, 1998.