Implicit Prioritisation via Weighted Pressure in Adaptive Traffic Signal Control

Tinaabishegan Baladewan
School of Computer Science
University of Nottingham Malaysia
Semenyih, Malaysia
tinaabishegan@gmail.com

Chee Jun Kit
School of Computer Science
University of Nottingham Malaysia
Semenyih, Malaysia
cheejk@gmail.com

***Abstract*— Urban traffic congestion is a significant challenge, and Reinforcement Learning (RL) based Adaptive Traffic Signal Control (ATSC) offers a potential solution. This paper presents a layout-specific ATSC approach tested in the SUMO simulator, using a Shared Noisy Double DQN agent with intersection grouping. While a generalisable graph neural network approach was considered, computational limits led to focusing on this method. Experiments in the Acosta and Pasubio scenarios demonstrated significant improvements over fixed-time control, reducing average waiting times by 17-25% and queue lengths by 11-17%. The study confirms that advanced DQN techniques combined with intersection grouping can create effective, computationally feasible ATSC systems for known network layouts.**

Keywords—Adaptive Traffic Signal Control, Reinforcement Learning, Deep Q-Network (DQN), SUMO, Intersection Grouping, Noisy Networks, Traffic Simulation, Intelligent Transportation Systems.

I. Introduction

Urban traffic congestion is a persistent global issue, causing significant economic and environmental costs, largely due to the limitations of traditional fixed-time traffic signal controllers [1]. These systems often fail to adapt to the dynamic nature of real-time traffic flow. Adaptive Traffic Signal Control (ATSC) offers a promising alternative by dynamically adjusting signal timings based on current traffic conditions to improve efficiency [2]. In recent years, Reinforcement Learning (RL), particularly Deep Reinforcement Learning (DRL), has emerged as a powerful data-driven approach for developing sophisticated ATSC strategies [1]. RL allows controllers (agents) to learn optimal policies through direct interaction with the traffic environment, often simulated using platforms like SUMO (Simulation of Urban MObility) [1]. This learning process typically involves an agent navigating states (traffic conditions), taking actions (signal changes), and receiving rewards based on traffic performance metrics [1].

This paper focuses on the implementation and evaluation of a layout-specific ATSC strategy within SUMO. We employ an advanced DRL agent based on a Shared Noisy Double Deep Q-Network (DQN) architecture. Key features include intersection grouping, where similar intersection layouts share a common control model to enhance efficiency and consistency, and the use of Noisy Networks for effective exploration. While generalisation techniques using Graph Neural Networks exist [3],[4],[5], this work concentrates on optimising performance for known network configurations, leveraging the strengths of DQN variants tailored to specific layouts.

A. Environment and Task Definition

The environment for this research is the Simulation of Urban MObility (SUMO) platform [1], a microscopic traffic simulator. The task is to develop and evaluate an RL agent capable of adaptively controlling traffic signals within a defined SUMO network configuration. The agent must learn a control policy that effectively manages signal phasing based on real-time traffic conditions, aiming to mitigate congestion and improve overall traffic flow efficiency compared to baseline control strategies.

B. Research Focus

The central research question addressed in this paper is: How effective is a Shared Noisy Double DQN agent, incorporating intersection grouping based on layout similarity, in learning an adaptive traffic signal control policy for a specific network layout within the SUMO simulation environment, when compared to traditional fixed-time control methods? The focus is specifically on the performance and adaptability achievable with this layout-specific DRL approach.

II. Literature Review

The application of RL to ATSC has seen rapid growth, resulting in a diverse landscape of algorithms and methodologies. Deep Q-Networks (DQN) served as a foundational technique, demonstrating improvements over traditional methods [1]. However, standard DQN faces challenges like Q-value overestimation [7]. Enhancements such as Double DQN, which decouples action selection and evaluation [7], and Dueling DQN, which separates state-value and action-advantage estimation [6], have been developed to improve stability and performance.

Effective exploration is crucial in complex traffic environments. Noisy Networks (NoisyNets) offer an alternative to simpler methods like epsilon-greedy by integrating learnable parametric noise into the network, enabling potentially state-dependent exploration strategies [8]. Implementations combining NoisyNets with Double and Dueling DQN have shown promise in ATSC [8].

State representation and reward function design are critical [7]. Common state features include queue length [1], [6], [9], [10], waiting time [1], [10], and traffic pressure [11], [12], while reward functions often aim to minimise delay [6] or queue length [10], or optimise throughput [10] or pressure-based metrics [11]. Balancing these competing objectives in the reward function is a key design challenge [13], [14].

As ATSC often involves coordinating multiple intersections, Multi-Agent Reinforcement Learning (MARL)

is frequently employed [15]. Parameter Sharing (PS), where agents share network weights, is a common technique to improve data efficiency and scalability in MARL [16], [7]. However, network heterogeneity can limit the effectiveness of global PS [16]. Intersection grouping addresses this by applying PS selectively within clusters of intersections deemed similar based on static or dynamic features [11], as demonstrated in frameworks like GPLight [11]. This grouping approach allows for specialised policies within homogeneous clusters while maintaining scalability [11].

While this paper focuses on a layout-specific approach optimised for a known environment, other research explores generalisation using techniques like Graph Neural Networks (GNNs) [1], [11], [3]–[5], transfer learning [13], [5], and zero-shot methods [3]–[5], seen in models like IG-RL [3], MuJAM [5], and TransferLight [4]. These aim for broader applicability across unseen networks but often involve increased complexity. Our layout-specific focus represents a pragmatic trade-off, aiming for high performance within a defined operational context.

III. Methodology

This section details the two distinct RL methodologies explored for ATSC within the SUMO environment. Approach 1 aimed for zero-shot generalisation using graph representations and curriculum learning, while Approach 2 focused on layout-specific adaptivity using advanced DQN techniques and intersection grouping. Due to computational limitations hindering the full development of Approach 1, this paper primarily implements and evaluates Approach 2. However, the methodology for Approach 1 is outlined here for completeness.

A. Learning Algorithm and Agent Architecture

Approach 1 (Zero-Shot Generalisation) utilised Proximal Policy Optimisation (PPO). It interacts with the environment to collect rollouts and computes advantages using Generalised Advantage Estimation (GAE) before performing policy and value function updates.

Approach 2 (Layout-Specific Adaptivity) This approach employed a Shared Noisy Double Deep Q-Network (DQN) agent. The "Noisy" component replaces traditional exploration strategies (like epsilon-greedy) with parametric noise added to the network's weights, allowing the agent to learn an exploration policy. The "Shared" aspect indicates that a single instance of the DQN agent's networks is shared across multiple intersections that are grouped based on layout similarity.

B. State Representation

Approach 1's environment state was encoded as a hierarchical graph. This graph consisted of multiple node types: lane segments (representing portions of lanes with features like vehicle density, positional encoding, signal status, speed, and vehicle type distribution), movements, and intersections. Edges connected nodes hierarchically (segment-to-movement, movement-to-intersection) and spatially (segment-to-segment, inter-lane).

For Approach 2, the state for each intersection was represented as a flat feature vector. This vector included normalised queue length for each incoming lane, normalised cumulative waiting time for each incoming lane, one-hot encoding indicating the currently active green phase, normalised time elapsed since the last phase change, normalised weighted pressure metric.

C. Action Space

Approach 1's PPO agent's policy network output logits, forming a categorical distribution over possible phase indices for an intersection. Actions were sampled from this distribution.

Approach 2's DQN agent outputs Q-values for each possible phase index. The action selected is typically the phase index with the highest Q-value. The action space is discrete, representing the choice of the next green phase.

D. Network Model Architecture

Approach 1's core model was the TemporalGATTransformer. This architecture first processed the spatial information within the hierarchical graph state using multiple Graph Attention Network layers (GAT). GAT layers allow nodes to attend to their neighbours, weighting their influence based on feature similarity, thus capturing local spatial dependencies. The node embeddings produced by the GAT layers were then treated as a sequence over a fixed history length. This sequence was passed through a standard Transformer Encoder, incorporating positional encoding, to model temporal dependencies. The final output from the Transformer was fed into separate linear heads to produce the policy logits and the state value estimate required by the PPO algorithm.

Approach 2 utilised a Dueling Deep Q-Network. This consists of shared initial fully connected layers followed by two separate streams: one estimating the state value and the other estimating the advantage for each action. These streams are then combined to produce the final Q-values. Noisy linear layers were used instead of standard linear layers to incorporate parametric noise for exploration.

E. Reward Function Design

Approach 1 used a log-distance reward function. For each vehicle approaching an intersection, a reward component was calculated as $-weight * log(1 + distance)$, where distance is the vehicle's distance from the intersection and weight is a parameter associated with the vehicle's type. The total reward for an intersection was the sum of these components over all vehicles on controlled lanes.

Approach 2 utilised a shaped reward function, calculated per intersection based on readily available SUMO metrics. It combines penalties for average waiting time per vehicle and average queue length per lane with a reward for average vehicle speed across incoming lanes, using fixed weights:

$$reward = (-0.4 \cdot avg_{wait} - 0.3 \cdot avg_{queue} + 0.3 \cdot avg_{speed})/10.0 \quad (1)$$

F. Adaptivity and Generalisation Strategies

Approach 1's focus: Generalisation:

- Zero-Shot Goal & Graph Representation: The primary aim was to train a single model capable of operating on different, unseen network layouts without retraining. The hierarchical graph state representation was fundamental to this, providing a structured way to encode varying topologies that GNNs could potentially process invariantly.

- Curriculum Learning: Training was designed to occur over a curriculum of scenarios with increasing complexity. This progression intended to force the agent to learn generalisable control principles rather than overfitting to one specific setup.

Approach 2's focus: Layout-Specific Adaptivity:

- Intersection Grouping: This is the core mechanism for handling multiple intersections efficiently within a specific network. Intersections with identical layouts share the parameters of a single DQN model.
- Adaptive Exploration: The Noisy Network exploration strategy is adaptive. The agent learns the noise parameters, and the implementation includes logic to potentially increase the noise magnitude if recent performance degrades significantly, prompting more exploration to escape potential local optima or adapt to changing dynamics.

Both approaches implemented implicit prioritisation by using weighted pressure component in the state as a proxy for lane importance when calculating pressure. This implicitly prioritises lanes expected to handle more or faster traffic but does not differentiate based on vehicle class.

IV. Experimental Design

Approach 1's training was structured around a curriculum. However, this approach faced significant computational challenges that prevented its completion within the project timeframe:

- *SUMO API Overhead:* The complex graph state representation required numerous TraCI API calls per simulation step to gather detailed information leading to slow simulation progression.
- *SUMO Performance:* SUMO simulations, particularly when controlled via TraCI, are single threaded, resulting in low CPU utilisation (observed ~7%) and limiting the speed of individual simulations.
- *Parallelisation Issues:* Attempts to parallelise training by running multiple SUMO instances using Python's multiprocessing introduced significant overhead, further slowing down the overall training process rather than accelerating it.

Approach 2's training was conducted episodically, with each episode simulating a fixed duration 3600 seconds. To expose the agent to varying conditions, dynamically generated route files providing mixed traffic loads were used for each training episode. This approach proved computationally feasible. Training and evaluation were successfully performed using two medium-sized, publicly available SUMO scenarios: "Acosta" and "Pasubio". The agent for each scenario was trained for 2000 episodes. During the primary training phase reported, all vehicle types were assigned an equal weight of 1.0 for the implicit prioritisation mechanism.

A. Evaluation Metrics

The performance of the trained RL agent (Approach 2) is evaluated against a baseline fixed-time controller using standard traffic metrics, averaged over the duration of evaluation episodes.

Primary Metrics:

- Average Waiting Time (AWT): The average time vehicles spend stationary.
- Average Queue Length (AQL): The average number of stationary vehicles per incoming lane.

Secondary Metrics: To assess the impact of virtual prioritisation specific metrics are analysed for different vehicle classes, particularly comparing buses against standard passenger vehicles. These tests involve running the trained model with modified prioritisation settings on test data containing both vehicle types.

- Bus Average Waiting Time: AWT calculated specifically for vehicles classified as buses.
- Bus Average Speed: Average speed calculated specifically for buses.
- Other Vehicles' Average Waiting Time: AWT calculated for all non-bus vehicles.
- Other Vehicles' Average Speed: Average speed calculated for all non-bus vehicles.
- Average Queue Length.

V. Results

This section presents the empirical results obtained from evaluating Approach 2. The evaluation is structured in two stages: first, assessing the overall performance improvement compared to a baseline fixed-time controller, and second, investigating the effects of adjusting prioritisation weights.

A. Overall Performance Comparison

The comparison is conducted on the two test scenarios, Acosta and Pasubio, using the primary evaluation metrics: Average Waiting Time (AWT) and Average Queue Length (AQL).

Acosta Scenario:
The RL agent demonstrated significant improvements over the fixed-time baseline in the Acosta scenario. The AWT was reduced from 908.42 seconds for the fixed-time controller to 677.13 seconds for the RL agent, representing a 25.46% improvement. Correspondingly, the AQL decreased from 10.3 vehicles per lane to 8.55 vehicles per lane, a reduction of 17.04%.

Table I: Acosta Fixed-Time vs ATSC results

Metric	Fixed-Time	ATSC	Improvement
Avg. Waiting Time (s)	908.42	677.13	25.46%
Avg. Queue Length	10.3	8.55	17.04%

Pasubio Scenario:
Similar improvements were observed in the Pasubio scenario. The RL agent reduced the AWT from 1112.31 seconds (fixed time) to 917.17 seconds, an improvement of 17.54%. The AQL decreased from 16.4 vehicles per lane to 14.45 vehicles per lane, an 11.90% reduction.

Table II: Pasubio Fixed-Time vs ATSC results

Metric	Fixed-Time	ATSC	Improvement
Avg. Waiting Time (s)	1112.31	917.17	17.54%
Avg. Queue Length	16.4	14.45	11.90%

B. Virtual Prioritisation Effects

To evaluate the adaptability of implicit prioritisation, tests were conducted by assigning different conceptual weights to buses versus other passenger vehicles during evaluation runs. The impact was assessed using the secondary metrics, comparing runs where buses were conceptually weighted 1x, 2.5x, 5x, 7.5x, and 10x relative to passenger vehicles.

Acosta Scenario:
A weight of 5x yielded the lowest Bus AWT (549.24s) and highest Bus Speed (1.02 m/s), compared to the baseline RL run (571.11s AWT, 0.94 m/s Speed). However, this coincided with slightly higher waiting times for other vehicles (617.09s vs 682.01s baseline) and a similar overall AQL (8.53 vs 8.55 baseline). Weights higher than 5x resulted in increased waiting times for all vehicles, suggesting a negative impact on efficiency when prioritisation is too extreme. The lowest overall AWT was achieved with a 2.5x weight (578.29s).

Table III: Acosta Virtual Prioritisation experiments' results

Metric	Bus Weightage				
	x1	x2.5	x5	x7.5	x10
Avg. Bus Waiting Time (s)	571.11	563.49	**549.24**	557.04	575.09
Avg. Others' Waiting Time (s)	682.01	**579.23**	617.09	592.87	677.45
Avg. Waiting Time (s)	677.13	**578.29**	613.91	591.09	673.02
Avg. Bus Speed (m/s)	0.94	0.96	**1.02**	0.99	0.96
Avg. Others' Speed (m/s)	1.82	**1.89**	1.88	1.87	1.81
Avg. Speed (m/s)	1.79	**1.85**	**1.85**	1.84	1.78
Avg. Queue Length	8.55	**8.09**	8.53	8.22	8.70

Pasubio Scenario:
A conceptual bus weight of 5x resulted in the lowest Bus AWT (715.03s vs 835.3s baseline) and highest Bus Speed (1.03 m/s vs 0.96 m/s baseline). This also corresponded to the lowest overall AWT (854.93s vs 917.17s baseline) and a minimal change in AQL (14.47 vs 14.45 baseline). Again, higher weights (7.5x, 10x) led to increased waiting times compared to the 5x case.

Table IV: Pasubio Virtual Prioritisation experiments' results

Metric	Bus Weightage				
	x1	x2.5	x5	x7.5	x10
Avg. Bus Waiting Time (s)	835.3	824.37	**715.03**	841.24	824.83
Avg. Others' Waiting Time (s)	918.99	927.62	**858.27**	897.85	875.93
Avg. Waiting Time (s)	917.17	925.27	**854.93**	896.55	874.79
Avg. Bus Speed (m/s)	0.96	0.9	**1.03**	0.91	0.90
Avg. Others' Speed (m/s)	1.01	0.97	**1.04**	0.99	0.99
Avg. Speed (m/s)	1	0.97	**1.04**	0.99	0.99
Avg. Queue Length	**14.45**	14.94	14.47	14.74	14.96

These results suggest that the implicit prioritisation mechanism can influence service levels for different vehicle types. A moderate conceptual prioritisation (5x) appeared beneficial for buses without significantly degrading overall performance. However, the effect is complex and scenario-dependent, highlighting the challenges of achieving fine-grained prioritisation.

VI. Discussion

The implemented layout-specific Approach 2 demonstrated effective adaptive traffic signal control in SUMO simulations, achieving significant reductions in average waiting times and queue lengths compared to the fixed-time baseline (Tables I, II). This effectiveness is attributed to the use of Double DQN for stable learning, Dueling architecture for efficient state representation, Noisy Network exploration

for finding effective actions, and adaptive sigma for managing changing traffic dynamics.

Tests applying conceptual weights to investigate virtual prioritisation within Approach 2 showed that this method can influence service levels, such as reducing bus waiting times with moderate prioritisation (up to 5x) (Tables III, IV). However, these effects were scenario-dependent, and excessive prioritisation negatively impacted overall efficiency. This highlights the potential advantage of the explicit, though computationally challenging, prioritisation mechanisms considered in Approach 1 for more direct control over service differentiation.

This study faces several limitations. A key constraint is the layout-specific nature of the implemented approach; models trained for intersections require retraining for significantly different network structures. Furthermore, computational performance limitations with SUMO restricted the evaluation of a more generalisable Approach 1, underscoring practical difficulties in developing widely applicable reinforcement learning agents using current simulation tools.

Additional limitations arise from Approach 2's reliance on local information, leading to a lack of awareness of neighbouring intersection conditions and potentially suboptimal network coordination. The metrics used, like pressure and queue length, can also be misleading under certain conditions. Finally, the agent's actions are confined to the phase combinations predefined in the SUMO scenario files, potentially excluding more optimal control strategies.

Approach 1, incorporating GNN, hierarchical representation, and temporal modelling, was designed to mitigate issues like poor coordination and limited state understanding inherent in purely local agents. Theoretically, its features allow for reasoning about spatial dependencies and traffic dynamics, potentially leading to better network-level control, though it proved computationally demanding.

A significant limitation is that evaluations were confined to simulations, leaving real-world applicability uncertain due to the sim-to-real gap. Furthermore, the implicit prioritisation method tested provides less fine-grained control compared to explicit approaches.

Despite these constraints, the results successfully address the research objective. The study demonstrated that the developed agent significantly outperforms fixed-time control for specific layouts within SUMO. This confirms that employing advanced DQN features alongside layout-based parameter sharing presents a viable and computationally feasible method for enhancing traffic flow in known network configurations.

VII. Conclusion

A. Reflection on Contributions and Limitations of the Work

This research successfully demonstrates an effective layout-specific adaptive traffic signal control system using a Shared Noisy Double DQN agent within SUMO simulations. The main contribution is the practical implementation showing significant reductions in average waiting times and queue lengths compared to fixed-time controls in the tested scenarios. Efficient management of multiple intersections was achieved through parameter sharing for identical layouts, while advanced DQN techniques enhanced the agent's learning and adaptability. The work also explored implicit prioritisation, indicating potential to influence service for specific vehicles like buses, though explicit methods are likely needed for precise control.

However, several limitations exist. The models are trained for specific network layouts, limiting direct transferability. Computational challenges with the simulation tools hindered the development of a more ambitious, generalisable graph-based approach. The implemented system relies on local information, restricting inter-intersection coordination and potentially leading to suboptimal network flow. Additional constraints include potential metric misinterpretations, action spaces limited by predefined simulation phases, and the inherent gap between simulation results and real-world performance.

B. Future Work

Based on the study's findings and identified limitations, several promising avenues for future research emerge. A primary direction involves revisiting the zero-shot generalisation strategy outlined in Approach 1. To overcome the computational bottlenecks encountered with SUMO, this approach, particularly the graph neural network model, could be transferred to a more modern and performant simulation platform like CityFlow. Its support for multithreading and potential for faster simulations might make training complex graph-based models over diverse scenarios feasible, which would be a significant step towards achieving truly generalisable adaptive traffic signal control agents.

Furthermore, improving coordination between intersections is crucial. Building upon the graph-based framework proposed in Approach 1, future work could explore hierarchical control structures. This concept might involve agents operating at different scales: local intersection agents managed by regional or sub-network coordination agents. Such a structure could allow for both immediate local responsiveness and broader network-level optimisation, explicitly addressing coordination issues like poorly timed green waves.

Refining prioritisation mechanisms also remains an important area. While the implicit methods explored showed some effect, integrating explicit, configurable weighting based on vehicle type would permit more precise traffic management policies, for example, favouring public transport or emergency vehicles. Additionally, further investigation into state representations that capture network topology and traffic dynamics more effectively, while remaining robust against potential metric misinterpretations, is needed. Finally, validating any developed approach, especially generalisable ones, across a wider range of simulation scenarios and bridging the crucial sim-to-real gap through testing with real-world data are essential steps towards practical deployment.

References

[1] X. Jia *et al.*, "Adaptive Traffic Signal Control Based on Graph Neural Networks and Dynamic Entropy-Constrained Soft Actor–Critic," *Electronics*, vol. 13, no. 23, p. 4794, 2024, doi: 10.3390/electronics13234794.

[2] R. Zhao *et al.*, "Sequence Decision Transformer for Adaptive Traffic Signal Control," *Sensors (Basel, Switzerland)*, vol. 24, no. 19, p. 6202, 2024, doi: 10.3390/s24196202.

[3] F.-X. Devailly, D. Larocque, and L. Charlin, "IG-RL: Inductive Graph Reinforcement Learning for Massive-Scale Traffic Signal Control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7496–7507, 2022, doi: 10.1109/TITS.2021.3070835.

[4] J. Schmidt, F. Dreyer, S. A. Hashimi, and S. Stober, "TransferLight: Zero-Shot Traffic Signal Control on any Road-Network," *arXiv preprint arXiv:2412.09719*, 2024. [Online]. Available: https://arxiv.org/abs/2412.09719

[5] Devailly, F. X., Larocque, D., & Charlin, L., "Model-Based Graph Reinforcement Learning for Inductive Traffic Signal Control," in *Proc. Conf. Robot Learn. (CoRL)*, 2024. [Online]. Available: https://www.researchgate.net/publication/378910992_Model-Based_Graph_Reinforcement_Learning_for_Inductive_Traffic_Signal_Control

[6] M. T. Rafique, A. Mustafa, and H. Sajid, "Reinforcement Learning for Adaptive Traffic Signal Control: Turn-Based and Time-Based Approaches to Reduce Congestion," *arXiv preprint arXiv:2408.15751*, 2024. [Online]. Available: https://arxiv.org/abs/2408.15751

[7] S. Wang *et al.*, "Deep Reinforcement Learning-Based Traffic Signal Control Using High-Resolution Event-Based Data," *Entropy (Basel, Switzerland)*, vol. 21, no. 8, p. 744, 2019, doi: 10.3390/e21080744.

[8] M. T. Rafique, M. Umair, I. U. Haq, and Z. H. Abbas, "Reinforcement Learning for Adaptive Traffic Signal Control: Turn-Based and Time-Based Approaches to Reduce Congestion," *arXiv preprint arXiv:2408.15751v2*, Aug. 2024. [Online]. Available: https://arxiv.org/abs/2408.15751v2

[9] C. Cai and M. Wei, "Adaptive urban traffic signal control based on enhanced deep reinforcement learning," *Scientific Reports*, vol. 14, art. no. 10165, 2024, doi: 10.1038/s41598-024-64885-w.

[10] M. Guo, P. Wang, C.-Y. Chan, and S. Askary, "A Reinforcement Learning Approach for Intelligent Traffic Signal Control at Urban Intersections," *arXiv preprint arXiv:1905.07698*, 2019. [Online]. Available: https://arxiv.org/abs/1905.07698

[11] Y. Liu *et al.*, "GPLight: Grouped Multi-agent Reinforcement Learning for Large-scale Traffic Signal Control," in *Proc. Thirty-Second Int. Joint Conf. on Artificial Intelligence (IJCAI-23)*, Macao, S.A.R. of China, 2023, pp. 199–207, doi: 10.24963/ijcai.2023/23.

[12] I. Arel, C. Liu, T. Urbanik, and A. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intelligent Transport Systems*, vol. 4, no. 2, pp. 128–135, 2010, doi: 10.1049/iet-its.2009.0070.

[13] D. N, H. R, P. R, S. S, and S. P, "Adaptive Traffic Control Using Deep Reinforcement Learning," in *2024 IEEE Punecon*, Pune, India, 2024, pp. 1-8, doi: 10.1109/PuneCon63413.2024.10895492.

[14] S. Wang and S. Wang, "A Novel Multi-Agent Deep RL Approach for Traffic Signal Control," *arXiv preprint arXiv:2306.02684*, 2023. [Online]. Available: https://arxiv.org/abs/2306.02684

[15] Y. Fu, L. Zhong, Z. Li, and X. Di, "Federated Hierarchical Reinforcement Learning for Adaptive Traffic Signal Control," *arXiv preprint arXiv:2504.05553*, 2025. [Online]. Available: https://arxiv.org/abs/2504.05553

[16] Y. Zhang *et al.*, "Unicorn: A Universal and Collaborative Reinforcement Learning Approach Towards Generalizable Network-Wide Traffic Signal Control," *arXiv preprint arXiv:2503.11488*, 2025. [Online]. Available: https://arxiv.org/abs/2503.11488