A Constrained Multi-Agent Reinforcement Learning Approach to Traffic Management

Anirudh Satheesh[1], **Keenan Powell**[1]

[1]University of Maryland, College Park
anirudhs@terpmail.umd.edu, kpowell1@terpmail.umd.edu

Abstract

Traffic congestion in modern cities is exacerbated by the limitations of traditional fixed-time traffic signal systems, which fail to adapt to dynamic traffic patterns. Adaptive Traffic Signal Control (ATSC) algorithms have emerged as a solution by dynamically adjusting signal timing based on real-time traffic conditions. However, the main limitation of such methods is they are not transferable to environments under real-world constraints, such as balancing efficiency, minimizing collisions, and ensuring fairness across intersections. In this paper, we view the ATSC problem as a constrained multi-agent reinforcement learning (MARL) problem and propose a novel algorithm named Multi-Agent Proximal Policy Optimization with Lagrange Cost Estimator (MAPPO-LCE) to produce effective traffic signal control policies. Our approach integrates the Lagrange multipliers method to balance rewards and constraints, with a cost estimator for stable adjustment. We also introduce three constraints on the traffic network: GreenTime, GreenSkip, and PhaseSkip, which penalize traffic policies that do not conform to real-world scenarios. Our experimental results on three real-world datasets demonstrate that MAPPO-LCE outperforms baseline MARL algorithms in all environments and traffic constraints. Our results show that constrained MARL is a valuable tool for traffic planners to deploy scalable and efficient ATSC methods in real-world traffic networks.

1 Introduction

Traditional traffic signal systems, which operate on pre-programmed, fixed schedules, are often inadequate in addressing the dynamic nature of urban traffic flow due to an inability to adapt to constantly changing traffic patterns. This can result in longer waiting times and unfair traffic distributions across intersections (Curtis 2017). To combat the limitations of traditional fixed-time traffic signal systems, Adaptive Traffic Signal Control (ATSC) methods have been developed to adjust signal timing based on real-time traffic conditions dynamically. However, while ATSC methods hold promise in reducing congestion in busy intersections, there are still uncertainties about their deployment in real-world environments. One challenge is balancing efficiency while minimizing vehicle collisions and other hazards (Essa and Sayed 2020). Another challenge is maximizing the fairness of each intersection, or ensuring that the green times (amount of time the current traffic light is green) for different lanes is the same on average (Raeis and Leon-Garcia 2021). In general, these challenges highlight the ongoing struggles with incorporating constraints into ATSC methods that accurately reflect the demands of real-world environments.

Previous works on ATSC use the observations of the intersections to form traffic control policies, such as SOTL (Cools, Gershenson, and D'Hooghe 2007). However, these are heuristic-based and cannot adapt to more complex traffic environments. Additionally, they do not consider how current actions can affect future states, which hinders long-term outcomes. Reinforcement Learning (RL) has also been used to develop autonomous traffic control methods by optimizing over current and future states (Zheng et al. 2019a). This includes actor-critic methods (Aslani, Mesgari, and Wiering 2017) and policy gradient methods (Mousavi, Schukat, and Howley 2017; Pang and Gao 2019) on single intersection (Oroojlooy et al. 2020) and multi-intersection environments (Wei et al. 2019b; Chen et al. 2020). RL has also been used for non-traditional intersections such as roundabouts (Rizzo, Vantini, and Chawla 2019) and dynamical lane changing systems (Zhou et al. 2022).

Due to the exponentially growing action space of reinforcement learning as the number of intersections increases, it becomes difficult to learn effective single-agent RL policies that can adapt to non-stationary environments like traffic signal control. As such, some works formulate ATSC as a decentralized Multi-Agent Reinforcement Learning (MARL) problem, using several agents to represent each intersection instead of one agent as a global traffic controller. This allows each intersection to act as its own local RL agent under partial observability and maximize its utility along with the global utility (Zhou et al. 2022; Wang et al. 2020; Wang and Wang 2023). Additional work serves to improve baseline MARL algorithms by improving sample efficiency (Huang, Wu, and Boulet 2023), or adding information to the state space to mitigate partial observability, such as communication methods (Jiang et al. 2022) and environment modeling (Bao et al. 2024; Wei et al. 2019a).

Due to the efficacy of MARL in solving high dimensional traffic control problems and current struggles with incorporating constraints that reflect real-world environments, we

propose a constrained MARL algorithm named Multi-Agent Proximal Policy Optimization with Lagrange Cost Estimator (MAPPO-LCE). Specifically, the algorithm uses the Lagrange multipliers method to balance the constraints with maximizing rewards and a cost estimator function to update the Lagrange multiplier.

Our contributions can be summarized as follows:

1. We define three constraint functions: GreenSkip, GreenTime, and PhaseSkip, which penalize policies that do not reflect real-world scenarios.
2. We propose a constrained MARL algorithm for multi-intersection traffic control.
3. We show experimentally that MAPPO-LCE outperforms three baseline MARL algorithms on three different datasets.
4. Our results show that constrained MARL can be a valuable tool for traffic planners to deploy ATSC methods in real-world traffic networks to reduce congestion.

2 Related Work

In this section, we discuss recent work on MARL algorithms and general constraints for ATSC.

2.1 MARL for ATSC

Recent work uses multi-agent reinforcement learning to model traffic signal control, with each agent controlling one intersection under partial observability. Wang et al. (2020) developed independent and joint Advantage Actor-Critic (A2C) algorithms for ATSC with a centralized critic in a distributed setting. Chen et al. (2021) also leverages A2C in a multi-agent setting, using decentralized critics for each agent in a distributed network. In addition to on-policy algorithms, previous works use multi-agent off-policy algorithms for ATSC. For example, Zhang et al. (2023) use Nash Q-Learning to alleviate the large state-action space from traditional MARL algorithms. Wang and Wang (2023) improves on this by using a Deep Q-Network (Mnih et al. 2013) with Friend Q-Learning (Littman 2001) to achieve better coordination between agents.

Other ways to improve MARL algorithms in ATSC are to include additional information in each agent's observation space to create more informed policies. However, including more information does not always lead to better results, as this can require more parameters and a slower convergence rate (Zheng et al. 2019b). Thus, selecting the right information to include between agents is crucial for performance. Huang, Wu, and Boulet (2023) use a model-based approach by learning a global probabilistic dynamics model along with the policy, which generates a prediction of the next states as additional information. This method is purely decentralized, where there is no interaction between agents. Thus, Jiang et al. (2022) develops UniComm, a method that computes only the necessary information between neighbor agents, which is used in their UniLight algorithm to calculate Q values for each agent.

2.2 Constraints for ATSC

Solving environments with incorporated constraints is difficult due to balancing rewards and costs from the constraints. Constrained Reinforcement Learning (CRL) is an active research area in RL that solves such environments by developing algorithms that exclusively learn policies that are both effective and satisfy the constraints (e.g. safety, fairness, etc.) (Gu et al. 2022; Achiam et al. 2017; Lu et al. 2021). Achiam et al. (2017) develops a Constrained Policy Optimization (CPO) algorithm to learn policies under constraints, and Gu et al. (2022) expands this into a multi-agent setting with MACPO and MAPPO-Lagrange. Tabas, Zamzam, and Zhang (2023) improve upon MACPO by developing a primal-dual optimization framework and parameterizing each agent with a neural network.

In ATSC, there is minimal work on incorporating constraints into the environment to develop policies closer to real-world scenarios. Gu et al. (2024) partitions the traffic network topology to alleviate scalability issues with MARL, but this only constrains the state space, not the action space. Haydari et al. (2024) use the CRL framework with the amount of emissions as the constraint and develop a Soft Actor-Critic algorithm to balance rewards with constraints. However, this is a single-agent setting, which poses scalability issues as the number of intersections increases. Adan et al. (2023) models traffic environment constraints in a multi-agent setting, but this work models agents as the vehicles around one intersection, instead of each intersection being an agent. Finally, Raeis and Leon-Garcia (2021) creates two fairness constraints for the ATSC problem, one delay-based metric which is meant to diminish the number of vehicles experiencing significantly longer waiting times and another throughput-based metric which attempts to give equal weighting to all traffic flows by extending concepts from computer networking. However, this is also a single agent setting in a more simplistic environment and is focused more specifically on fairness between the North-South and East-West traffic flows instead of general constraints.

3 Preliminaries

In this section, we define the Constrained Markov Game, the RL environment, and our constraints for ATSC.

3.1 Constrained Markov Game for ATSC

We can model ATSC as a constrained Markov Game (Qu, Ma, and Wu 2024; Wang et al. 2024) which can be represented by the tuple $M = \langle \mathcal{N}, S, \{O_i\}_{i\in\mathcal{N}}, \{A_i\}_{i\in\mathcal{N}}, \mathcal{T}, r, \Omega, C, c, \gamma \rangle$, where $\mathcal{N} = \{1, 2, ..., n\}$ is a set of n agents; S is the state space; $O = \times_{i\in\mathcal{N}} O_i$ is the joint observation space, where O_i is the observation space of agent i; $A = \times_{i\in\mathcal{N}} A_i$ is the joint action space, where A_i is the action space of agent i; $T : S \times A \times S \to [0, 1]$ is probabilistic state transition function; R is the reward function; $\Omega : S \times A \times O \to [0, 1]$ is space of conditional observation probabilities ($\Omega(s', a, o) = P(o|s', a)$); $C : S \times A \to \mathbb{R}$ is the cost function; and c is the cost limit. Since this is a decentralized Markov Game, the reward function for each

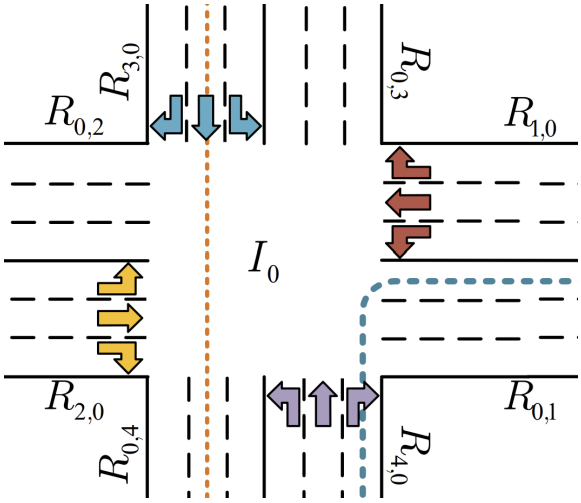

Figure 1: An example intersection with three lanes for turning left, going straight, and turning right for each of the four incoming directions/roads.

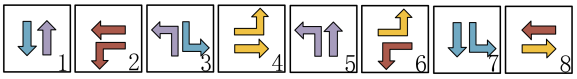

Figure 2: An overview of the 8 different phases for a four-direction intersection.

agent is the same, e.g. $R = R_i \, \forall i \in \mathcal{N}$. MARL algorithms for constrained Markov Games aim to search for policy π that solves this constrained optimization problem:

$$\max_{\pi} \; \mathbb{E}_{(s_t \sim S, a_t \sim \pi)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right],$$
$$\text{s.t.} \quad \mathbb{E}_{(s_t \sim S, a_t \sim \pi)} \left[\sum_{t=0}^{\infty} \gamma^t C(s_t, a_t) \right] < c$$

In the ATSC problem specifically, the elements of the environment are defined as:

- Agents: Each agent is responsible for controlling traffic lights at one intersection.
- Observation: The observation of each agent is composed of the characteristics of the corresponding intersection. Specifically, each intersection has 12 road links (vehicles turning left, right, and going straight in each cardinal direction), and each road link contains the number of vehicles moving, the number of vehicles waiting, the traffic light phase, and the number of vehicles in each lane, as well as the speed and location of each vehicle in the lane.
- Actions: As shown in Figure 1 and Figure 2, there are eight phases that describe combinations of traffic lights that can be green simultaneously. At each timestep, the intersection can choose one of these phases as an action.
- STATE: The state is the combination of all observations at the current time step.
- STATE Transition: After an action is selected at each time step, vehicles are allowed to move if the corresponding traffic light is green for a short period T_g. While the environment does not directly represent yellow lights, before changing phases, all lights that would be turned on/off are turned to red for a brief period T_y before the lights of the new phase are turned to green.
- Reward: Each agent will receive a global reward $\lambda_f R_f + \lambda_w R_w$, where R_f is the total number of vehicles moving, R_w is the total number of vehicles waiting, and λ_f and λ_w are hyperparameters.

For more information on environment parameters, refer to Appendix A.

3.2 Environment Constraints

We develop three environment constraints on each intersection that reflect real-world environments named GreenTime, PhaseSkip, and GreenSkip (German Road and Transport Research Association 2010). These constraints also help to promote fair treatment of all vehicles by the agents by reducing differences in waiting times between directions and encouraging agents to take all possible actions.

- GreenTime: Each light l should be green for no more than $G_{max\ time}$ before turning red to prevent long waiting times from other lanes, and model light cycles in the real world. Each time step that a light is on increases its GreenTime value by 1, and when it is turned off its GreenTime value is set to 0. Right-turn lights are ignored for this constraint, as they are always treated as being on.

$$G_{time}(l) \leq G_{max\ time} \tag{1}$$

- PhaseSkip: The state of each traffic light follows one of a specific, pre-determined set of phases (see Figure 2). No phase should be skipped consecutively more than $P_{max\ skip}$ times. Each time the phase changes, the new phase has its PhaseSkip value set to 0, and all phases other than the new phase and the old phase have their PhaseSkip values incremented by 1. This is a way of somewhat closely approximating how traffic cycles work in the real world, as well as being an indirect way of promoting the agent to give equal attention to all lanes.

$$P_{skips}(p) \leq P_{max\ skips} \tag{2}$$

- GreenSkip: Similar to the phase constraint, no individual light l should be skipped consecutively more than $G_{max\ skips}$ times. Each time the phase changes, each light turned on in the new phase has its GreenSkip value set to 0, and all lights not on in the new phase or the old phase have their GreenSkip values incremented by 1. This is a direct way of promoting fairness by reducing the variance in waiting times among all lanes. Right-turn lights are also ignored for this constraint.

$$G_{skips}(l) \leq G_{max\ skips} \tag{3}$$

Each agent is constrained according to Eqns 1-3. The penalty associated with each constraint is the average across

all lights:

$$\frac{\sum_{i\in\mathcal{N}}\sum_{l}\frac{1_c}{n_l(i)}}{|\mathcal{N}|} \tag{4}$$

where 1_c is an indicator function that checks whether the constraint is satisfied, i is the intersection, $|\mathcal{N}|$ is the number of agents, l is a specific light at the intersection the agent controls, and $n_l(i)$ is the total number of lights at the intersection that particular agent controls. Note that for the PhaseSkip constraint, we sum over the phases and divide by the total number of phases. For our experiments, the number of lights is always 12 and the number of phases is always equal to 8, as all the intersections in our environment have 4 roads of 3 lanes with 8 distinct phases. For more information on constraints, including the algorithm for each constraint, refer to Appendix A.

4 Method

In this section, we describe our constrained multi-agent reinforcement learning algorithm: Multi-Agent Proximal Policy Optimization with Lagrange Cost Estimator (MAPPO-LCE).

4.1 Multi-Agent Proximal Policy Optimization with Lagrange Cost Estimator

Constrained optimization problems are typically of the form

$$\max_x f(x)$$
$$\text{s.t. } g(x) \leq c$$

which can be solved by the Lagrange multiplier method

$$\mathcal{L}(x;\lambda) = f(x) - \lambda(g(x) - c) \tag{5}$$

where $\mathcal{L}(x;\lambda)$ is a new optimization objective to maximize and $\lambda > 0$ is the Lagrange multiplier. (Bertsekas 1996). Thus, for the constrained MARL problem,

$$\max_{\pi_\theta} \ \mathbb{E}_{(s_t\sim S, a_t\sim\pi_\theta)}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\right], \tag{6}$$

$$\text{s.t.} \quad \mathbb{E}_{(s_t\sim S, a_t\sim\pi_\theta)}\left[\sum_{t=0}^{\infty}\gamma^t C(s_t,a_t)\right] < c \tag{7}$$

we can formulate it with a Lagrangian where

$$f(x) = \mathbb{E}_{(s_t\sim S, a_t\sim\pi)}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\right] \tag{8}$$

$$g(x) = \mathbb{E}_{(s_t\sim S, a_t\sim\pi)}\left[\sum_{t=0}^{\infty}\gamma^t C(s_t,a_t)\right] \tag{9}$$

In MAPPO-LCE, we use a reward critic and a cost critic, $V^r_{\phi_r}$ and $V^c_{\phi_c}$, for estimating the discounted cumulative reward and discounted cumulative cost, respectively. Instead of training on every step, we also collect a dataset D containing rollout data every episode: $\{s_t, r_t, c_t, s_{t+1}\}$. After B episodes, we update the policy. Similar to MAPPO-Lagrange (Gu et al. 2022), we aim to minimize the following loss:

$$\mathcal{L}(\pi_\theta) = \mathcal{L}_r(\pi_\theta) - \lambda\mathcal{L}_c(\pi_\theta) \tag{10}$$

where $\mathcal{L}_r$ and $\mathcal{L}_c$ are the MAPPO (Yu et al. 2022) actor losses with an unclipped critic loss term:

$$\begin{aligned}\mathcal{L}_r(\pi_\theta) =& \mathbb{E}_{s_t\sim D, a_t\sim\pi_\theta}\left[\min\left(\rho_t A^r_t, \text{clip}(\rho_t, 1\pm\epsilon)A^r_t\right)\right] \\ &+ \beta\mathbb{E}_{s_t\sim D}\left[\frac{1}{2}\|V^r_{\phi_r}(s_t) - r_t\|^2\right]\end{aligned} \tag{11}$$

$$\begin{aligned}\mathcal{L}_c(\pi_\theta) =& \mathbb{E}_{s_t\sim D, a_t\sim\pi_\theta}\left[\min\left(\rho_t A^c_t, \text{clip}(\rho_t, 1\pm\epsilon)A^c_t\right)\right] \\ &+ \beta\mathbb{E}_{s_t\sim D}\left[\frac{1}{2}\|V^c_{\phi_c}(s_t) - c_t\|^2\right]\end{aligned} \tag{12}$$

In these formulations, ρ_t is the importance sampling ratio

$$\rho_t = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$$

A^c_t and A^r_t are the cost advantage and reward advantage functions respectively, and ϵ is the clipping parameter. Here, we abuse notation and say that $A^c_t = A^c_t(s_t,a_t)$ and $A^r_t = A^r_t(s_t,a_t)$. We also update each critic by the temporal difference error (TDE):

$$\mathcal{L}_{\phi_r} = \mathbb{E}_{(s_t,\, s_{t+1})\sim D}\left[r_t + \gamma V^r_{\phi_r}(s_{t+1}) - V^r_{\phi_r}(s_t)\right] \tag{13}$$

$$\mathcal{L}_{\phi_c} = \mathbb{E}_{(s_t,\, s_{t+1})\sim D}\left[c_t + \gamma V^c_{\phi_c}(s_{t+1}) - V^c_{\phi_c}(s_t)\right] \tag{14}$$

In MAPPO-Lagrange (Gu et al. 2022), the Lagrange multiplier λ is updated by the advantage function $A^c_t(s_t,a_t)$. However, since the advantage function can be unstable and potentially incorrectly estimate the constraint, we incorporate a Langrage Cost Estimator inspired by Qu, Ma, and Wu (2024). This cost estimator learns the cost dynamics to accurately predict the cost, and then updates λ. We train the cost estimator θ_C by minimizing the following loss:

$$\mathcal{L}_{\theta_C} = \|\theta_C(s_t,a_t) - c_t\|^2,\ s_t\sim D, a_t\sim\pi_\theta \tag{15}$$

Finally, we update λ with the following loss to ensure that the constraint function is satisfied under the cost limit c:

$$\mathcal{L}_\lambda = \mathbb{E}_{s_t\sim D, a_t\sim\pi_\theta}\left[-\lambda(\theta_C(s_t,a_t) - c)\right] \tag{16}$$

as the loss is minimized when the estimated cost is much less than the cost limit.

5 Experiments

5.1 Environment Setup

We run our experiments on MAPPO-LCE and related baselines on the CityFlow environment (Zhang et al. 2019), which is a scalable and realistic traffic simulator due to its C++ backend. Additionally, it is compatible with several multi-agent RL algorithms by integrating with the Gymnasium library (Brockman 2016). From Wei et al. (2019b), there are three publicly available datasets collected from real-world traffic data from Hangzhou, China (HZ); Jinan, China (JN); and New York, USA (NY). Details of each environment are located in Table 1.

To evaluate the performance of each of the MARL algorithms, we use three evaluation metrics:

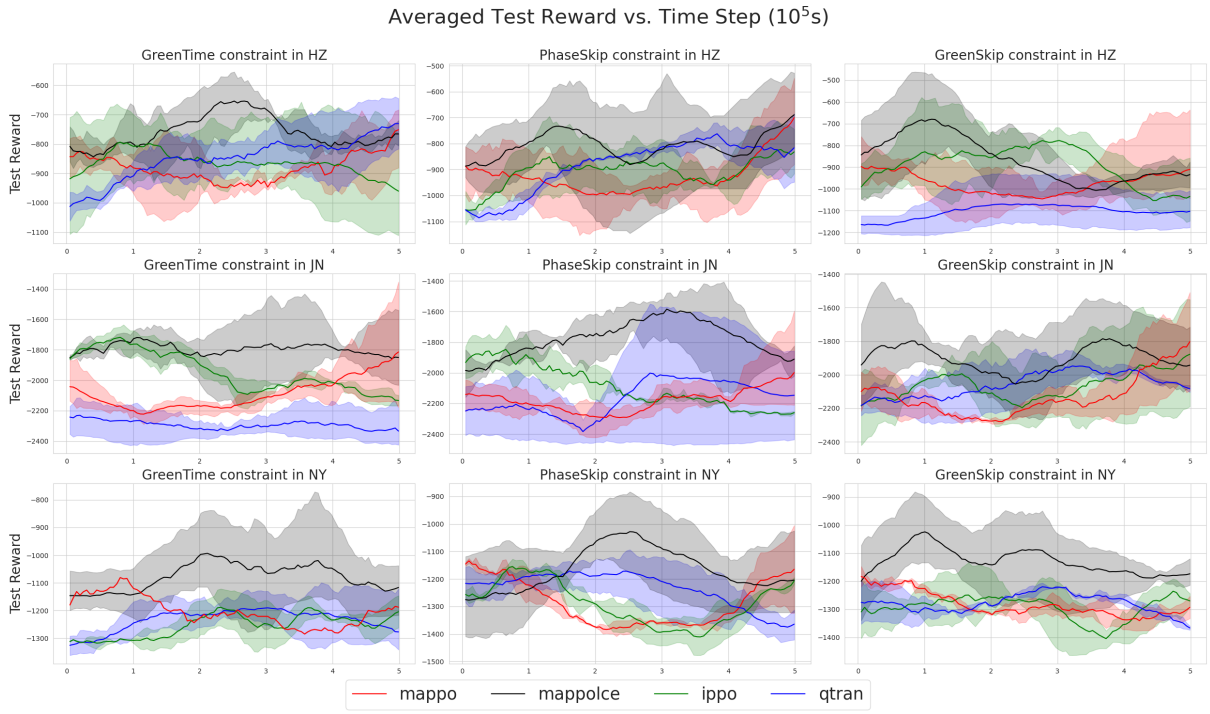

Figure 3: Test Reward results of our algorithm and baseline algorithms across all environments and constraints.

Algorithm 1: MAPPO-LCE Algorithm

Initialize replay buffer $\mathcal{D}$, policy parameters θ, critic networks V_ϕ^r, V_ϕ^c, cost network θ_C, and Lagrange multiplier λ.

for each episode **do**

  for each time step t **do**

    Select action $a_t = \pi_\theta(s_t)$

    Execute joint action a_t at state s_t

    Observe reward r_t, cost c_t, and next state s_{t+1}

    $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, r_t, c_t, s_{t+1})$

  end for

  Sample batch $\mathcal{B}$ from $\mathcal{D}$

  $\theta \leftarrow \theta - \alpha \nabla \mathcal{L}(\pi_\theta)$

  $\phi^r \leftarrow \phi^r - \alpha \nabla \mathcal{L}_{\phi_r}$

  $\phi^c \leftarrow \phi^c - \alpha \nabla \mathcal{L}_{\phi_c}$

  $\theta_C \leftarrow \theta_C - \alpha_{\theta_C} \nabla \mathcal{L}_{\theta_C}$

  $\lambda \leftarrow \lambda - \alpha_\lambda \nabla_\lambda \mathcal{L}_\lambda$

  Clamp to ensure $\lambda \geq 0$

  Soft update actor and critic parameters:

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta', \quad \phi' \leftarrow \tau\phi + (1-\tau)\phi'$$

end for

- Test Reward: The test reward is the same as the training reward: $\lambda_f R_f + \lambda_w R_w$.
- Average Delay: The average delay is the average delay across all vehicles, which is the total travel time minus the expected travel time for each vehicle. The expected travel time is the estimated time the vehicle should finish its route if there were no traffic lights.
- Throughput: The throughput of the environment is the number of vehicles that complete their routes before the episode ends.

	HZ	JN	NY
Number of Intersections	16	12	48
Number of Lanes	3	3	3
Total Number of Vehicles	2983	6295	2824
Time Steps (s)	3600	3600	3600

Table 1: Summary of Traffic Metrics for HZ, JN, and NY.

5.2 Baseline Methods

In this work, we compare our algorithm to three baseline MARL algorithms: Independent Proximal Policy Optimization (IPPO) (De Witt et al. 2020), Multi-Agent Proximal Policy Optimization (MAPPO) (Yu et al. 2022), and QTRAN (Son et al. 2019). This set of algorithms allows us to test both on-policy algorithms (IPPO, MAPPO), and off-policy algorithms (QTRAN).

- IPPO (De Witt et al. 2020): IPPO treats each agent as its independent local RL agent to maximize local rewards. This transforms the problem into $|\mathcal{N}|$ independent single-agent PPO rollouts.

- MAPPO (Yu et al. 2022) MAPPO joins the actions of each agent into a single joint action vector, and each agent shares an actor network and a critic network to update the policy.

- QTRAN (Son et al. 2019): QTRAN develops an unstructured value function factorization, which allows for more generalizable decentralized execution of MARL problems.

All baseline algorithms were implemented from the ePYMARL library (Papoudakis et al. 2021). For each of the baseline experiments, the total reward at time step t is $r_t - \zeta c_t$, where r_t and c_t are the rewards and costs at time step t, and ζ is a hyperparameter that trades off maximizing the reward and satisfying the constraints. All experiments were conducted on a single RTX A5000 GPU.

6 Results

The results of the algorithms on the three environments are shown in Figure 3. In these figures, we include the results of each algorithm on the test reward function defined in section 5.1. In Appendix B, we show the results on the average delay and throughput metrics.

As shown in the figure, our proposed MAPPO-LCE algorithm outperforms the baselines in almost all environment settings. In the HZ dataset, IPPO achieves average test rewards of -794, -823, -778 on the GreenTime, PhaseSkip, and GreenSkip constraints respectively. Generally, the performance of QTRAN and MAPPO perform better on the GreenTime and PhaseSkip constraints than IPPO, and perform worse on the GreenSkip constraint. On the harder JN dataset (we consider the JN dataset to be harder than the HZ datasetdue to the increased number of cars, even if there are fewer intersections), MAPPO-LCE continues to outperform the baseline algorithms with average test rewards of -1720, -1584,-1785. On the most complex NY dataset (we consider the NY dataset to be the most complex due to the large number of intersections), our algorithm performs the best comparably, with high scores of -993, -1027, and -1024 on GreenTime, PhaseSkip, and GreenSkip respectively. The next highest scores are all from MAPPO and are -1080, -1133, -1183 respectively, a significant improvement by our algorithm across all constraints.

We note that MAPPO-LCE maintains performance increases relative to the baseline algorithms as the dataset difficulty increases, as taking the average test reward difference between the best and second-best algorithms yields performance gains of 8.24%, 5.09%, and 10.8% on the HZ, JN, and NY datasets respectively. This underscores the scalability of our algorithm, since as the environment difficulty increases, our algorithm can find optimal policies that can balance satisfying the constraints while maintaining effectiveness.

7 Future Work

For future work, one idea is to incorporate communication between agents, which in this case are the traffic signals. This would allow traffic lights to communicate information such as the number of vehicles going from one intersection to another, the average time it takes to get there, etc. vital information that could help discover more optimal policies. This also can be easily represented with an underlying Graph Neural Network (GNN), that can pass messages from the state space along its edges and can update via backpropagation, which means we can add message loss to the overall loss as both the GNN and the neural networks in each algorithm use backpropagation.

Another idea is to expand our constraints. Formulating our current constraints as hard constraints and adding additional soft constraints such as variance in throughput or waiting times could more closely represent real-world environments while creating a model that values fairness and safety. In addition, we could add further constraints by expanding the environment to include different types of vehicles, such as buses, ambulances, and trams, to develop more generalizable traffic management strategies that accommodate diverse transportation needs. Adding more significant constraints on their delay and waiting time could lead to more robust constraints that better reflect real-life scenarios.

While ATSC is a partially observable Markov Game, a final idea is to give each agent a better idea of their surroundings through expectation alignment. ELIGN (Ma et al. 2022) is a method for multi-agent expectation alignment that aligns the shared expectations of an agent to its actual actions through an intrinsic reward. Predicting neighboring agents' actions in a second-order theory of mind approach allows for better coordination, and can easily be added to existing methods to find more optimal policies. In the ATSC problem, this may allow each agent to predict swells or dips in traffic before they reach the intersection that the agent controls, further increasing its ability to make fair and safe decisions.

8 Conclusion

In this paper, we focus on finding scalable algorithms for the Adaptive Traffic Signal Control problem in real-world traffic environments. We propose a novel algorithm, MAPPO-LCE for constrained multi-agent reinforcement learning. We expand upon Multi-Agent Proximal Policy Optimization (MAPPO) by incorporating elements of MAPPO-Lagrangian (Gu et al. 2022) and introducing a Lagrange Cost Estimator to accurately predict constraints even under unstable conditions. While we only focused on three constraints, MAPPO-LCE can be used with any number of general traffic constraints, and can be extended to any constrained MARL problem. Our experimental results using the CityFlow environment in multiple real-world settings show that MAPPO-LCE outperforms other baseline methods with suitable constraints. Our findings indicate that constrained multi-agent reinforcement learning can identify more optimal traffic policies for ATSC in real-world conditions and holds strong potential for real-world deployment.

References

Achiam, J.; Held, D.; Tamar, A.; and Abbeel, P. 2017. Constrained policy optimization. In *International conference on machine learning*, 22–31. PMLR.

Adan, F.; Feng, Y.; Angeloudis, P.; Quddus, M.; and Ochieng, W. 2023. Constrained Multi-Agent Reinforcement Learning Policies for Cooperative Intersection Navigation and Traffic Compliance. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, 4079–4085.

Aslani, M.; Mesgari, M. S.; and Wiering, M. 2017. Adaptive traffic signal control with actor-critic methods in a real-world traffic network with different traffic disruption events. *Transportation Research Part C: Emerging Technologies*, 85: 732–752.

Bao, T.; Wei, H.; Ji, J.; Work, D.; and Johnson, T. T. 2024. Spatial-Temporal PDE Networks for Traffic Flow Forecasting. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track: European Conference, ECML PKDD 2024, Vilnius, Lithuania, September 9–13, 2024, Proceedings, Part X*, 166–182. Berlin, Heidelberg: Springer-Verlag. ISBN 978-3-031-70380-5.

Bertsekas, D. P. 1996. *Constrained Optimization and Lagrange multiplier methods*. Athena Scientific.

Brockman, G. 2016. OpenAI Gym. *arXiv preprint arXiv:1606.01540*.

Chen, C.; Wei, H.; Xu, N.; Zheng, G.; Yang, M.; Xiong, Y.; Xu, K.; and Li, Z. 2020. Toward A Thousand Lights: Decentralized Deep Reinforcement Learning for Large-Scale Traffic Signal Control. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04): 3414–3421.

Chen, Y.; Li, C.; Yue, W.; Zhang, H.; and Mao, G. 2021. 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)*, 1–6.

Cools, S.-B.; Gershenson, C.; and D'Hooghe, B. 2007. *Self-Organizing Traffic Lights: A Realistic Simulation*, 41–50. Springer London. ISBN 9781846289828.

Curtis, E. 2017. EDC-1: Adaptive Signal Control Technology — Federal Highway Administration.

De Witt, C. S.; Gupta, T.; Makoviichuk, D.; Makoviychuk, V.; Torr, P. H.; Sun, M.; and Whiteson, S. 2020. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*.

Essa, M.; and Sayed, T. 2020. Self-learning adaptive traffic signal control for real-time safety optimization. *Accident Analysis & Prevention*, 146: 105713.

German Road and Transport Research Association. 2010. *RiLSA: Guidelines für Traffic Signals. Traffic Lights for Road Traffic.* Köln: FGSV-Verlag.

Gu, H.; Wang, S.; Ma, X.; Jia, D.; Mao, G.; Lim, E. G.; and Wong, C. P. R. 2024. Large-Scale Traffic Signal Control Using Constrained Network Partition and Adaptive Deep Reinforcement Learning. *Trans. Intell. Transport. Sys.*, 25(7): 7619–7632.

Gu, S.; Kuba, J. G.; Wen, M.; Chen, R.; Wang, Z.; Tian, Z.; Wang, J.; Knoll, A.; and Yang, Y. 2022. Multi-Agent Constrained Policy Optimisation. arXiv:2110.02793.

Haydari, A.; Aggarwal, V.; Zhang, M.; and Chuah, C.-N. 2024. Constrained Reinforcement Learning for Fair and Environmentally Efficient Traffic Signal Controllers. *ACM J. Auton. Transport. Syst.*, 2(1).

Huang, X.; Wu, D.; and Boulet, B. 2023. Fairness-Aware Model-Based Multi-Agent Reinforcement Learning for Traffic Signal Control.

Jiang, Q.; Qin, M.; Shi, S.; Sun, W.; and Zheng, B. 2022. Multi-Agent Reinforcement Learning for Traffic Signal Control through Universal Communication Method. In Raedt, L. D., ed., *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, 3854–3860. ijcai.org.

Littman, M. L. 2001. Friend-or-Foe Q-learning in General-Sum Games. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, 322–328. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 1558607781.

Lu, S.; Zhang, K.; Chen, T.; Başar, T.; and Horesh, L. 2021. Decentralized policy gradient descent ascent for safe multi-agent reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 8767–8775.

Ma, Z.; Wang, R.; Li, F.-F.; Bernstein, M.; and Krishna, R. 2022. Elign: Expectation alignment as a multi-agent intrinsic reward. *Advances in Neural Information Processing Systems*, 35: 8304–8317.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing Atari with Deep Reinforcement Learning. arXiv:1312.5602.

Mousavi, S. S.; Schukat, M.; and Howley, E. 2017. Traffic Light Control Using Deep Policy-Gradient and Value-Function Based Reinforcement Learning. arXiv:1704.08883.

Oroojlooy, A.; Nazari, M.; Hajinezhad, D.; and Silva, J. 2020. AttendLight: Universal Attention-Based Reinforcement Learning Model for Traffic Signal Control. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 4079–4090. Curran Associates, Inc.

Pang, H.; and Gao, W. 2019. Deep Deterministic Policy Gradient for Traffic Signal Control of Single Intersection. In *2019 Chinese Control And Decision Conference (CCDC)*, 5861–5866.

Papoudakis, G.; Christianos, F.; Schäfer, L.; and Albrecht, S. V. 2021. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS)*.

Qu, Y.; Ma, J.; and Wu, F. 2024. Safety Constrained Multi-Agent Reinforcement Learning for Active Voltage Control. arXiv:2405.08443.

Raeis, M.; and Leon-Garcia, A. 2021. A Deep Reinforcement Learning Approach for Fair Traffic Signal Control. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2512–2518. IEEE Press.

Rizzo, S. G.; Vantini, G.; and Chawla, S. 2019. Reinforcement Learning with Explainability for Traffic Signal Control. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 3567–3572.

Son, K.; Kim, D.; Kang, W. J.; Hostallero, D. E.; and Yi, Y. 2019. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*, 5887–5896. PMLR.

Tabas, D.; Zamzam, A. S.; and Zhang, B. 2023. Interpreting Primal-Dual Algorithms for Constrained Multiagent Reinforcement Learning. In *Learning for Dynamics and Control Conference*, 1205–1217. PMLR.

Wang, S.; and Wang, S. 2023. A Novel Multi-Agent Deep RL Approach for Traffic Signal Control. arXiv:2306.02684.

Wang, T.; Liang, T.; Li, J.; Zhang, W.; Zhang, Y.; and Lin, Y. 2020. Adaptive Traffic Signal Control Using Distributed MARL and Federated Learning. In *2020 IEEE 20th International Conference on Communication Technology (ICCT)*, 1242–1248.

Wang, Z.; Fang, M.; Tomilin, T.; Fang, F.; and Du, Y. 2024. Safe Multi-agent Reinforcement Learning with Natural Language Constraints. *arXiv preprint arXiv:2405.20018*.

Wei, H.; Chen, C.; Zheng, G.; Wu, K.; Gayah, V.; Xu, K.; and Li, Z. 2019a. PressLight: Learning Max Pressure Control to Coordinate Traffic Signals in Arterial Network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, 1290–1298. New York, NY, USA: Association for Computing Machinery. ISBN 9781450362016.

Wei, H.; Xu, N.; Zhang, H.; Zheng, G.; Zang, X.; Chen, C.; Zhang, W.; Zhu, Y.; Xu, K.; and Li, Z. 2019b. CoLight: Learning Network-level Cooperation for Traffic Signal Control. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, 1913–1922. New York, NY, USA: Association for Computing Machinery. ISBN 9781450369763.

Yu, C.; Velu, A.; Vinitsky, E.; Gao, J.; Wang, Y.; Bayen, A.; and Wu, Y. 2022. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35: 24611–24624.

Zhang, H.; Feng, S.; Liu, C.; Ding, Y.; Zhu, Y.; Zhou, Z.; Zhang, W.; Yu, Y.; Jin, H.; and Li, Z. 2019. Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario. In *The world wide web conference*, 3620–3624.

Zhang, Y.; Wang, S.; Ma, X.; Yue, W.; and Jiang, R. 2023. Large-Scale Traffic Signal Control by a Nash Deep Q-network Approach. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, 4584–4591.

Zheng, G.; Xiong, Y.; Zang, X.; Feng, J.; Wei, H.; Zhang, H.; Li, Y.; Xu, K.; and Li, Z. 2019a. Learning Phase Competition for Traffic Signal Control. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, 1963–1972. New York, NY, USA: Association for Computing Machinery. ISBN 9781450369763.

Zheng, G.; Zang, X.; Xu, N.; Wei, H.; Yu, Z.; Gayah, V.; Xu, K.; and Li, Z. 2019b. Diagnosing Reinforcement Learning for Traffic Signal Control. arXiv:1905.04716.

Zhou, W.; Chen, D.; Yan, J.; Li, Z.; Yin, H.; and Ge, W. 2022. Multi-agent reinforcement learning for cooperative lane changing of connected and autonomous vehicles in mixed traffic. *Autonomous Intelligent Systems*, 2(1).

A Hyperparameter Selection

A.1 Algorithm Hyperparameters

Since all of our baseline algorithms are taken from the ePYMARL library (Papoudakis et al. 2021), we use the same hyperparameters. For the constraint trade-off hyperparameter $zeta$, we set it to 0.2 for all constraints. We model the cost estimator as a Multi-Layer Perceptron (MLP), with two hidden layers and a hidden layer size of 128. We also use an Adam optimizer with a learning rate of 10^{-4} to train the cost estimator. We set the cost limit for all experiments to 0.

A.2 Environment Hyperparameters

Each time step in the environment is composed of T_g inner steps to update the environment, which we set to 10. After the policy selects an action, each inner step simulates 1 second of the environment. To simulate yellow lights without actually implementing them directly, each traffic light instead turns off all lights that would be switched between phases for T_y time before fully turning them red or green. We set T_y to 5 time steps, which is equivalent to 5 seconds. Each constraint also has a hyperparameter that controls its severity. For constraint thresholds, we set $G_{max\ time}$ to 40, $P_{max\ skips}$ to 16, and $G_{max\ skips}$ to 4. The specific algorithm for how each constraint is calculated is in Algorithm 2.

B Further Results

Figures 4 and 5 below show the results for the average delay and average throughput metrics, defined in Section 5.1. Each algorithm's objective is to minimize the delay and maximize the throughput. The figures show our algorithm continues to outperform the baseline methods in most environments under most constraints, particularly in the NY and JN environments. While our algorithm still outperforms QTRAN and MAPPO in the HZ environment, IPPO either outperforms or is nearly equal to our algorithm on these metrics.

Algorithm 2: Constraint Calculation

```
for time = 1 to N do
    ...
    {GreenTime Calculation}
    for light in lights do
        if light on in current phase then
            green_time[light] += 1
        else
            green_time[light] = 0
        end if
    end for
    ...
    {PhaseSkip Calculation}
    if new_phase ≠ old_phase then
        for phase in phases do
            if (phase ≠ old_phase) and (phase ≠ new_phase)
            then
                phase_skips[phase] += 1
            end if
        end for
        phase_skips[new_phase] = 0
        ...
        {GreenSkip Calculation}
        for light in lights do
            if (light red in old_phase) and (light red in
            new_phase) then
                green_skips[light] += 1
            else
                green_skips[light] = 0
            end if
        end for
    end if
    ...
end for
```

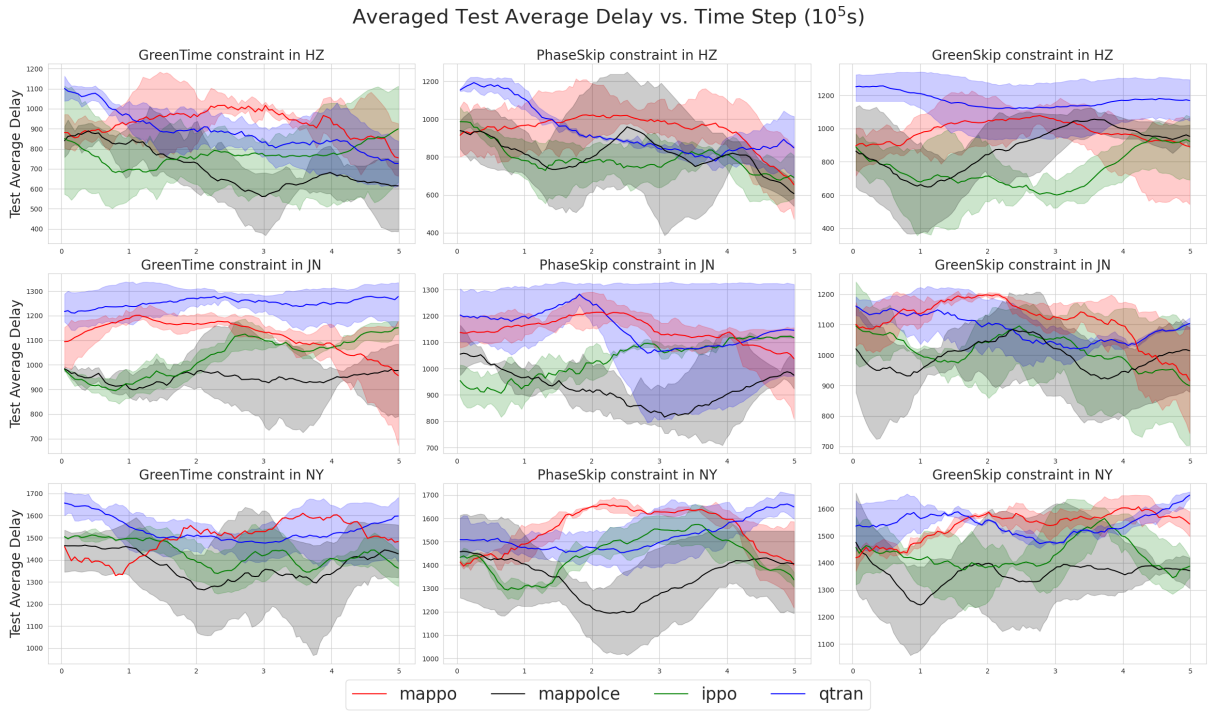

Figure 4: Average Delay results of our algorithm and baseline algorithms across all environments and constraints (lower is better).

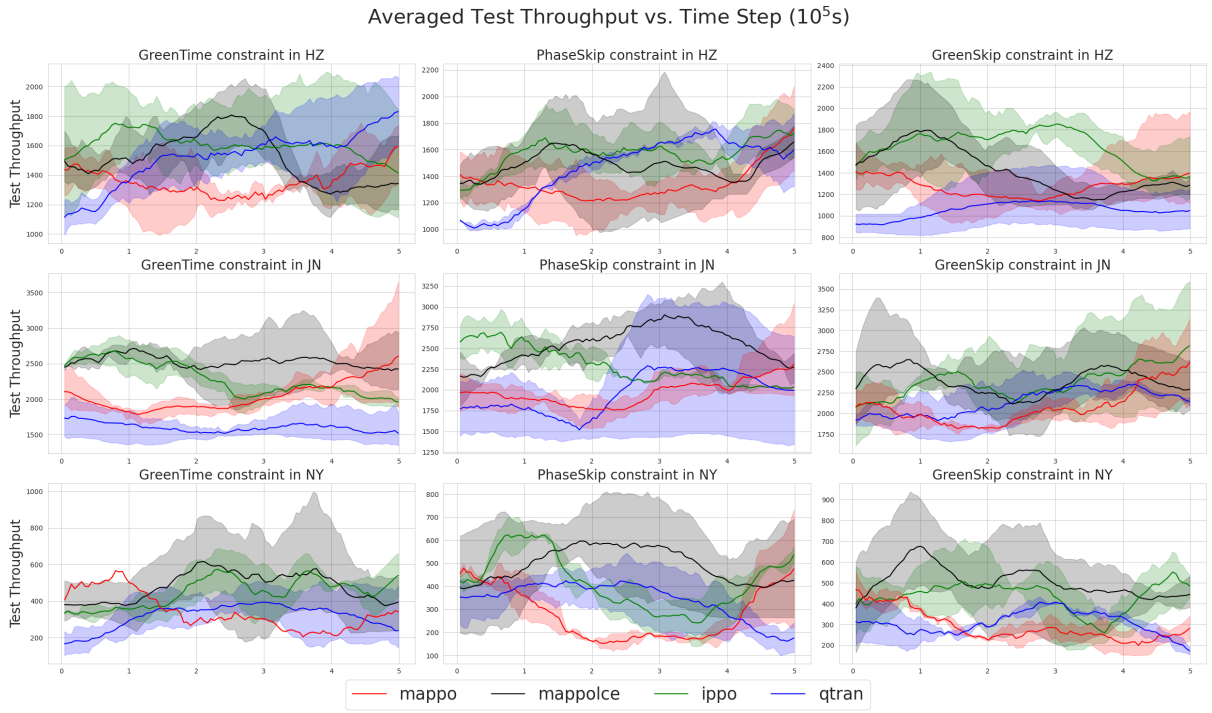

Figure 5: Throughput results of our algorithm and baseline algorithms across all environments and constraints.