AlphaRoute: Large-Scale Coordinated Route Planning via Monte Carlo Tree Search

Guiyang Luo [1, 2*], **Yantao Wang**[1*], **Hui Zhang**[3], **Quan Yuan**[1, 2†], **Jinglin Li**[1]

[1] State Key Laboratory Of Networking And Switching Technology (Beijing University of Posts and Telecommunications)
[2] State Key Laboratory of Integrated Services Networks (Xidian University)
[3] Beijing Jiaotong University, Beijing, China

Abstract

This paper proposes AlphaRoute, an AlphaGo inspired algorithm for coordinating large-scale routes, built upon graph attention reinforcement learning and Monte Carlo Tree Search (MCTS). We first partition the road network into regions and model large-scale coordinated route planning as a Markov game, where each partitioned region is treated as a player instead of each driver. Then, AlphaRoute applies a bilevel optimization framework, consisting of several region planners and a global planner, where the region planner coordinates the route choices for vehicles located in the region and generates several strategies, and the global planner evaluates the combination of strategies. AlphaRoute is built on graph attention network for evaluating each state and MCTS algorithm for dynamically visiting and simulating the future state for narrowing down the search space. AlphaRoute is capable of 1) bridging user fairness and system efficiency, 2) achieving higher search efficiency by alleviating the curse of dimensionality problems, and 3) making an effective and informed route planning by simulating over the future to capture traffic dynamics. Comprehensive experiments are conducted on two real-world road networks as compared with several baselines to evaluate the performance, and results show that AlphaRoute achieves the lowest travel time, and is efficient and effective for coordinating large-scale routes and alleviating the traffic congestion problem. The code will be publicly available.

Introduction

With the emerging trend towards urbanization and the rapid development of urban transportation, traffic congestion is becoming a serious and one of the most urgent and vexing problems in metropolitan areas (Liang et al. 2022; Luo et al. 2022c; Zhang et al. 2022). Travelers complain about traffic congestion because it adds to their travel times a delay that can be used for other activities. For society and the environment, traffic congestion means more energy and money consumption, and increased greenhouse gas emission and air pollution (Zhou et al. 2017). Taking Beijing as an example, the average daily congestion duration in 2017 was 2.6 hours, while it was only 0.9 hours in 2009. The estimated direct or indirect economic losses caused by traffic jams to American car owners reach 300 billion, with an average of 1400 per driver. How to reduce traffic congestion is the main problem faced by the governments in many countries (Luo et al. 2022b).

Recent years have witnessed significant attention being attracted to traffic congestion, from both academia and industry (Dai et al. 2022). Municipalities continually invest in road infrastructure construction and maintenance to increase supply, although controversies on whether more roads alleviate congestion persist, i.e., Braess's Paradox (Frank 1981), which shows that adding capacity to a network can sometimes actually slow down the traffic (Çolak, Lima, and González 2016). Other efforts to reduce congestion aim to decrease driving demand by promoting alternative travel modes (Zhou et al. 2021), high occupancy driving lanes, carpooling (Gollapudi, Kollias, and Panigrahi 2019), congestion pricing (Chen et al. 2020), etc. However, these methods would bring about inconvenience to drivers and travelers.

Recently, drivers are increasingly leveraging real-time information through GPS devices and online routing tools (e.g., Google maps) to move faster using vehicular networks (Luo et al. 2022a, 2018, 2021). This trend offers a new tool to guide drivers to make choices for the benefit of the city, thus creating a more optimal traffic configuration (Su et al. 2022). Consequently, coordinated route planning (CRP) stands out as a promising and popular way to overcome the traffic congestion problem, i.e. considering the whole system welfare and assigning routes to users in such a way that the congestion problem is solved or, at least, kept at a minimum level (Morandi 2021).

Related Works and Challenges

However, coordinating large-scale routes is challenging due to the following challenges.

1. **A huge gap between the user equilibrium and the system optimum.** The well-known Wardropian principles, i.e., the user equilibrium (UE) and the system optimum (SO), are fundamental objectives for CRP (Zhang, Liu, and Waller 2019). UE guarantees fairness among users, in which each user chooses the most convenient path selfishly. When UE is reached, all users sharing the same origin and destination will experience the same travel time. SO involves overall efficiency, which is a system-wide

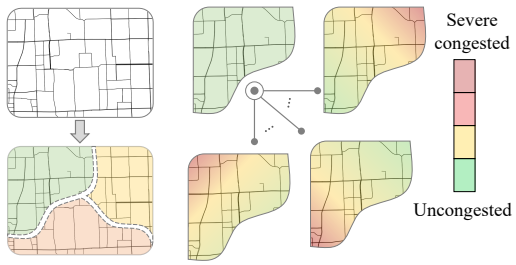
*These authors contributed equally.
†Corresponding author.
Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

traffic assignment that minimizes the total travel time. However, UE is fair but the total travel time is not minimized, and SO is the most efficient assignment while being unstable since it is unfair, resulting in users' non-compliance to the assigned routes (Klein, Levy, and Ben-Elia 2018). The gap between UE and SO is also known as "price of anarchy", which has been theoretically investigated by (Belov et al. 2021; Colini-Baldeschi et al. 2020; Correa, Cristi, and Oosterwijk 2021). However, how to reduce this gap in a practical and large-scale CRP is still missing.

2. **Complexity in coordinating massive routes.** The CRP problem has high computation complexity. Assume that n drivers are planning their routes at the same time, and each trip may have k potential routes to connect the source and destination, there exist k^n matching possibilities. Instead of direct search, pruning techniques are widely applied to reduce search space. Li *et al.* (Li, Chen, and Shang 2021) applied depth-first search with a pruning technique to enable early termination of the search. Chen *et al.* (Chen et al. 2020) developed an efficient and effective approximate matching algorithm based on local search, as well as pruning techniques to further enhance the matching efficiency. However, these approaches directly consider all possible routes for each driver and the complexity is still unsatisfactory since there exist hundreds of thousands of drivers in a city.
3. **Dynamic impact on the successive traffic situation.** Transportation traffic experiences spatio-temporal dynamics. In the spatial perspective, congestion at roads inside the city center or an area having popular venues like a stadium or hospital can cause ripples throughout large-scale transportation networks. In the temporal perspective, the congestion across different timestamps is changing and evolving. Consequently, current good traffic assignments would get worse in successive traffic situations. To handle this problem, Ma *et al.* (Ma et al. 2015) exploited a deep learning model for predicting traffic congestion evolution. Anwar *et al.* (Anwar et al. 2018) applied a two-layer approach to capture the evolution by incrementally updating the partitions in an efficient manner, where each partition has a homogeneous level of congestion. These methods exploit historical patterns for modeling and predicting congestion, which is statistical average rather than real-time optimizing.

Our Contributions

Inspired by AlphaGo (Silver et al. 2016), we propose AlphaRoute to coordinate the large-scale route planning, efficiently addressing the above challenges. We first partition the road networks into M regions and further propose a bilevel optimization framework, consisting of M region planners and a global planner. The region planner considers user equilibrium, and coordinates the route choices for vehicles within the region. The global planner is built on value function that adopts graph attention networks for evaluating each state (a combination of strategies) and Monte Carlo Tree Search (MCTS) (Fu, Qiu, and Zha 2021), which



(a) Partition (b) Possible strategies

Figure 1: (a) Partition the road network into M regions. (b) For each region m, the local planner generates several possible strategies for the vehicles $\mathcal{N}^m$.

can dynamical search the state space and simulate the future traffic for sufficient long steps. AlphaRoute is capable of achieving the following targets:

1. **Bridging the user equilibrium and system optimum.** AlphaRoute exploits the region planner to make coordinated traffic assignments in each region, which can reach user equilibrium. Then, a global planner can evaluate the combinations of strategies obtained by the regions, thus optimizing the total travel time. Consequently, AlphaRoute considers the whole system welfare, where the congestion problem is solved or, at least, kept at a minimum level, as well as individual fairness, where the users are willing to comply with the system's routing advice.
2. **Alleviating the curse of dimensionality.** Traditional methods based on the Markov game treat each driver as a player, and each player has an exponential number of paths from the source to destination. Therefore, it suffers from the curse of dimensionality, i.e., explosions in state and action spaces, making it challenging for practical application. To settle down this problem, AlphaRoute reformulates CRP as a modified Markov game, which significantly reduces the state-action space since each region planner is treated as a player. A region planner generates several strategies under different assignment objectives, where each strategy is a user equilibrium. All generated strategies form the action space for that region. Consequently, the action and state space are significantly decreased, alleviating the curse of dimensionality problem.
3. **Simulation over the future to capture traffic dynamics.** Instead of depending on the current traffic situation to coordinate route planning, AlphaRoute can simulate and evaluate the future traffic for sufficient long steps, and apply the simulation results of several steps to make an informed and effective decision. During the simulation over future states, AlphaRoute exploits graph attention reinforcement learning and MCTS algorithm to dynamically evaluate states for narrowing down the search space.

To evaluate the performance of AlphaRoute, comprehensive experiments are conducted on two real-world road networks. Compared with several baselines, AlphaRoute achieves the best performance in terms of travel time and

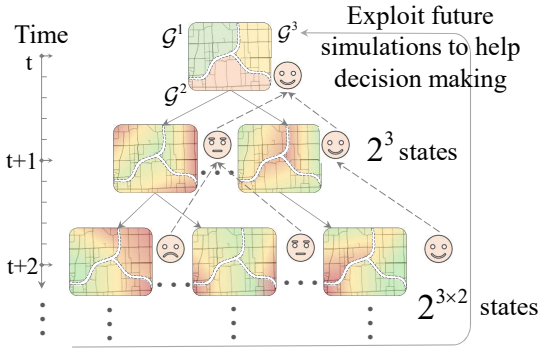


Figure 2: The proposed bilevel optimization framework, consisting of M region planners and a global planner.

waiting time. Furthermore, ablation studies are conducted to evaluate the performance of value function and Monte Carlo tree search.

Overview

Road networks. A road network can be described as a connected and directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the vertex set $\mathcal{V}$ means road intersections, and edge set $\mathcal{E}$ represents road segments. Each edge $e = v_i \rightarrow v_j \in \mathcal{E}$ connects two road intersections v_i and v_j, where $v_i, v_j \in \mathcal{V}$.

Markov game. On top of the road networks $\mathcal{G}$, a set of drivers $\mathcal{N}$ is supposed to make cooperative and coordinated route planning, where each driver $i \in \mathcal{N}$ makes a choice of route p_i with origin location o_i to destination d_i. We model such problem as a Markov game, denoted as a tuple $(\mathcal{S}, \mathcal{N}, \{\mathcal{P}^1, \cdots, \mathcal{P}^N\}, \mathcal{F}, \{q^1, \cdots, q^N\})$, where $N = |\mathcal{N}|$. The state space $\mathcal{S}$ counts the road network, and status of all drivers; the joint action space $\{\mathcal{P}^1, \cdots, \mathcal{P}^N\}$ represents the available actions for drivers at a state $s \in \mathcal{S}$; the state transition probability function $\mathcal{F}$ pushes current state s_t to a new state s_{t+1} after a joint action $\{p_1 \in \mathcal{P}^1, \cdots, p_N \in \mathcal{P}^N\}$ is taken; the reward q_t^i is negatively correlated to the travel time for driver $i \in \mathcal{N}$ at the end of the game (time slot T) and is zero if the game is not terminated. The large-scale routing planning is supposed to bridge the user equilibrium and system optimum, which are introduced as follows.

- **User equilibrium:** each driver $i \in \mathcal{N}$ behaves competitively in minimizing its own travel time, which is also known as fairness.
- **System optimum:** the total travel time of all drivers is at a minimum, which requires all users to behave cooperatively in choosing their routes to ensure the most efficient use of the transportation system, which is also known as efficiency.

The above Markov game models each driver as a player, and has been widely adopted in coordinated traffic assignment (Shou and Di 2020; Li, Zheng, and Yang 2019; Yu, Yu, and Gu 2019; Rahili et al. 2018; Koh et al. 2020; Zhao et al. 2021; Delarue, Anderson, and Tjandraatmadja 2020; Sun et al. 2019; Shou and Di 2020; Wang et al. 2020; Kang et al. 2021). However, it is computationally intensive and faces the curse of dimensionality problems: explosions in state and action spaces and high stochasticity i.e., a large number of possible next states for a given state-action pair. Therefore, it is difficult to obtain a scalable and efficient solution, not to mention bridging the user equilibrium and system optimum. To this end, we partition the road network into several regions and each region is regarded as a player, generating a new modified Markov game, which would reduce the action and state space.

Road network partition. The road network $\mathcal{G}$ is spatially partitioned into M parts, i.e., $\mathcal{G} = \mathcal{G}^1 \cup \mathcal{G}^2 \cup \cdots \cup \mathcal{G}^M$, with $\mathcal{G}^m = (\mathcal{V}^m, \mathcal{E}^m)$, where $\mathcal{V} = \mathcal{V}^1 \cup \mathcal{V}^2 \cup \cdots \cup \mathcal{V}^M$ and $\mathcal{V}^i \cap \mathcal{V}^j = \emptyset$, if $i \neq j$. Many graph partitioning algorithms can be exploited and this paper adopts the normalized graph cut algorithm presented in (Ji and Geroliminis 2012). Taken Beijing Sub-district, Beijing, China as an example, the corresponding road network is partitioned into 3 regions, which are shown in Fig.1(a).

Overview of the proposed framework. We propose a bilevel optimization framework, consisting of M region planners and a global planner. Our framework applies sequence decision-making, i.e., time is divided into time slots $\{t, t+1, \cdots, T\}$ and decisions are made for each time slot. We introduce these two planners as follows.

- **Region planner:** for each region m, a region planner coordinates the route choices of vehicles within this region $\mathcal{N}^m$, eventually output several possible strategies, where each strategy contains the route choices of all vehicles in $\mathcal{N}^m$, which is shown in Fig.1(b). This planner takes fairness into consideration.
- **Global planner:** this planner balances and evaluates the inter-region influence, and exploits simulations over the future to help decision-making. Since each region has got several possible strategies, this planner evaluates the combination strategies of all regions. Furthermore, exploiting digital twin-based ultra-high fidelity simulations, this planner simulates traffic state for the future steps $\{t+2, t+3, \cdots\}$ and adopts the simulated results for making informed decisions. However, the number of combination states is exponential with regions number and depth of simulations, therefore, we exploit the Monte Carlo tree search algorithm to achieve higher efficiency.

Modified Markov game. Each region is treated as a super player and M super players work together for coordinating the route choices of all drivers $\mathcal{N}$. Such problem can be modeled as a modified Markov game, denoted as a tuple $(\mathcal{S}, \mathcal{M}, \{\mathcal{A}^1, \cdots, \mathcal{A}^M\}, \mathcal{P}, \{r^1, \cdots, r^M\})$. $\mathcal{M}$ is the set of super players, i.e., all partitioned regions, $\mathcal{M} = \{1, 2, \cdots, M\}$. The joint action space $\{\mathcal{A}^1, \cdots, \mathcal{A}^M\}$ represents the available actions for regions $\mathcal{M}$ at a state $s \in \mathcal{S}$, where $\mathcal{A}^m = \bigcup_{i \in \mathcal{N}^m} \mathcal{P}^i$; the state transition probability function $\mathcal{P}$ pushes current state s_t to a new state s_{t+1} after a joint action $\{a^1, \cdots, a^M\}$ is taken, where $a^m = \bigcup_{i \in \mathcal{N}^m} p_i$; $\{r_t^1, \cdots, r_t^M\}$ is the rewards, $r_t^m = \sum_{i \in \mathcal{N}^m} q_t^i$.

We explain the route planning framework with the example shown in Fig.2. The city is partitioned into 3 regions. At time step t, the local planner obtains 2 possible strategies for each region. Our framework simulates the future traffic and

these exist 2^3 states at $t+1$ time step and $2^{3\times l}$ states at $t+l$ time step. For each state, the global planner evaluates each state (e.g., the state is good if less congested) and further exploits the evaluation for narrowing down the search space. After simulation and evaluation for sufficient long steps, our framework provides insights to make an informed and effective decision at the current time step, i.e., the simulation results of several steps to help the local planner to select the best from all possible strategies.

Methodology

The user equilibrium describes a state where every single driver has no gains to unilaterally deviate from the chosen route under a certain objective. However, the user equilibrium is proved to be unique and it does not conform with system optimum (Morandi 2021). In order to bridge this gap, we resort to user equilibrium under different objectives, where each objective leads to a user equilibrium.

Region Planner: Game Theory

The local planner m aims to coordinate the route choices of vehicles $\mathcal{N}^m$, which can be described as a game $(\mathcal{G}^m, \mathcal{N}^m, (\mathcal{P}^i)_{i\in\mathcal{N}^m}, (g_i)_{i\in\mathcal{N}^m})$. g_i is the payoff (cost) function. The strategy space $\mathcal{P}^i$ of a player $i \in \mathcal{N}^m$ is the set of all simple paths in $\mathcal{G}$ from o_i' to d_i', where o_i' and d_i' are the mapped locations of o_i and d_i on $\mathcal{G}^m$, respectively. Denote a strategy profile as $\boldsymbol{P} = (p_1, p_2, \cdots, p_{|\mathcal{N}^m|})$, where $p_i \in \mathcal{P}^i$. The payoff function can be set by considering different objectives, and are introduced as follows:

- **Objective A:** minimum free-flow travel latency (the travel time when there are no other vehicles), i.e.,

$$\min \sum_{i\in\mathcal{N}^m} g_i(\boldsymbol{P}) = \min \sum_{i\in\mathcal{N}^m} \sum_{e\in p_i} \frac{len(e)}{mxs(e)}, \quad (1)$$

  where $len(e)$ and $mxs(e)$ are the length and maximum speed of road segment e, respectively.

- **Objective B:** minimum travel distance, i.e.,

$$\min \sum_{i\in\mathcal{N}^m} g_i(\boldsymbol{P}) = \min \sum_{i\in\mathcal{N}^m} \sum_{e\in p_i} len(e). \quad (2)$$

- **Objective C:** minimum number of traffic lights, i.e.,

$$\min \sum_{i\in\mathcal{N}^m} g_i(\boldsymbol{P}) = \min \sum_{i\in\mathcal{N}^m} \sum_{e\in p_i} 1, \quad (3)$$

  where each road segment is assumed to be scheduled by a traffic light.

- **Objective D:** travel time with mutual influence. i.e.,

$$\min \sum_{i\in\mathcal{N}^m} g_i(\boldsymbol{P}) = \min \sum_{i\in\mathcal{N}^m} \sum_{e\in p_i} f_e(\boldsymbol{P}), \quad (4)$$

  where $f_e(\boldsymbol{P})$ means the actual travel time on e and is related to the number of vehicles using e.

Definition 1: user equilibrium. A strategy profile $(p_1, \cdots, p_i, \cdots, p_{|\mathcal{N}^m|})$ is a user equilibrium, if for any player $i \in \mathcal{N}$ choosing a different strategy p_i', its cost would not be decreased, i.e.,

$$g_i(p_1, \cdots, p_i, \cdots, p_N) \leq g_i(p_1, , \cdots, p_i', \cdots, p_N).$$

In such a case, no player has profitable unilateral deviations, i.e., no player alone can decrease their cost by switching to a different strategy.

Theorem 1: For each objective defined above, there exists one and only one user equilibrium.

Proof. The proof can be found in our Appendix. □

The strategy of the former three objectives does not consider the mutual influence between vehicles, which can be easily computed using an optimization algorithm under utility. As for the last one (objective D), we apply a polynomial-time algorithm presented in (Ravindran Vijayalakshmi and Skopalik 2020) for achieving the user equilibrium. Therefore, the predefined four objectives would output 4 different user equilibrium for each region. We need simulation of several steps, and there exist an exponential number of states, which we apply a value function to dynamically evaluate each state and MCTS algorithm for improving efficiency.

Global Planner: Monte Carlo Tree Search

The global planner combines the graph attention reinforcement learning in MCTS algorithm that selects actions by lookahead search, similarly to AlphaGo (Silver et al. 2016). However, the region planner $m \in \mathcal{M}$ applies different objectives for coordinating the route choices of vehicles $\mathcal{N}^m$, eventually obtaining several actions, where each action is a user equilibrium under an objective. The possibility of each action is difficult to define. Therefore, we ignore the prior probability of action and only consider the value function for each state in the lookahead search.

Each state s of the search tree stores an value $V(s)$ that maps the status of current traffic situation to a real value, and visit count $N(s)$ that records the number of accessing times. Starting from the root state, the tree is traversed by simulation. At each time step of the simulation, an action is selected from state s_t according to

$$a_t = argmax_a(V(s_{t+1}) + u(s_{t+1})). \quad (5)$$

The action is selected for maximizing the value function plus a bonus $u(s)$, which decays with repeated visits to encourage exploration and is calculated as

$$u(s) \propto \frac{1}{1+N(s)}. \quad (6)$$

When the traversal reaches a leaf node s_L at step L, the leaf node needs to be expanded. However, as mentioned earlier, it is impossible to evaluate the probability of the actions computed by the local planner. We fully expand the leaf node s_L to s_{L+1}. This step would output an exponential number of states, which can be refined using a pruning mechanism to keep at most n_{sta} states for each layer.

The leaf node s_L is evaluated in two very different ways: first, by the value function $v(s_L)$; and second, by the outcome z_L of a random rollout played out until the terminal

step using the fast rollout policy, where each player applies the shortest path for making a routing choice. We apply a mixing parameter λ for combining these two evaluations, i.e.,

$$V(s_L) = (1-\lambda)V(s_L) + \lambda z_L, \tag{7}$$

At the end of the simulation, the value function and visit counts of all traversed states are updated. Each sate accumulates the visit count and mean evaluation of all simulations passing through that state, i.e.,

$$N(s) = \sum_{i=1}^{H} 1(s,i), \tag{8}$$

$$V(s) = \frac{1}{N(s)} \sum_{i=1}^{H} 1(s,i) V(s_L^i), \tag{9}$$

where s_L^i is the leaf node from the i-th simulation, H is the total number of simulations, and $1(s,i)$ indicates whether the state s was traversed during the i-th simulation. Once the search is complete, the algorithm chooses the best state from the root position.

AlphaRoute consists of five steps, which is detailed narrated in our appendix.

Value Function: Graph Attention Network

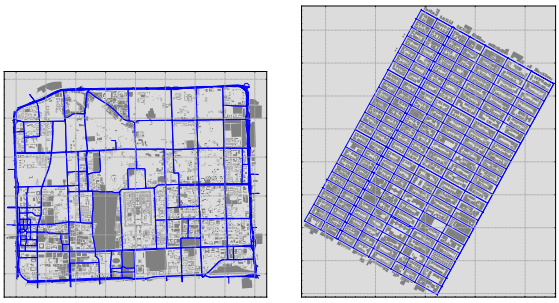
For each state s, the value function $V(s)$ evaluates the status of current traffic situation, which maps the state s into a real value, i.e., a bigger value means that current state has a higher probability of achieving smaller total travel time. The state s contains the road network topology with fine-grained road attributes, and real-time traffic, i.e., the information of all vehicles. The state can be represented as a graph, where the vertex means the road way. Each edge is assigned with a weight representing the distance between the connected vertexes, which is computed using thresholded Gaussian kernel (Li et al. 2017). The features of each vertex contain fine-grained road attributes, i.e., length, speed limit, direction, lanes, as well as vehicle flow statistic information on the road way, i.e., vehicle number, average speed, speed variance. Then, graph attention network (Veličković et al. 2017) is exploited to approximate the value function. During experiments, we have conduct extensive experiments for selecting the best network architecture, and finally we select a graph attention networks with 4 layers. We adopt the similar training strategy as in AlphaGo to conduct model-free reinforcement learning for training the value function.

Experiment

We conduct comprehensive experiments on two real-world road networks to evaluate AlphaRoute, especially the MCTS of the global planner which evaluates and chooses the combinations of actions taken by the region planners.

Experimental Settings

The effectiveness and strengths of AlphaRoute are validated on two real-world road networks, i.e., Beijing Sub-district, Beijing, China and Upper East Side, Manhattan, New York, USA. For each city, the corresponding road network is collected from OpenStreet Maps. Road networks are described by intersections, road segments (which are connections between intersections), and the fine-grained attributes of road segments. The road network is shown in Fig.3 and detailed information and comparison between these two maps are shown in Tab.1. The experiments are conducted utilizing a well-known open-source simulator SUMO. There exists three types of vehicles for conforming to the actual situation to the greatest extent, i.e., bus, truck, and car, with $10\%, 20\%,$ and 70% respectively.



(a) Beijing, China (b) Manhattan, New York, USA

Figure 3: Road networks from real-world datasets.

City maps	Beijing	Manhattan
#Road segments	783	463
Total road length (km)	338.55	59.47
#Intersections	309	214
Area (km^2)	35.89	2.7
Road attributes	Length, speed limit, lanes, etc.	

Table 1: Detailed information of road network.

There exist up to $5,000$ vehicles on these two maps. The vehicles are randomly located on the roads. 30% of the vehicles are initialized at the beginning and the rest are activated in the first 10 minutes. The destination d_i of each vehicle is randomly selected to guarantee that the distance between the origin location o_i and d_i is not too close, and this settings is fixed over all experiments. The time-varying traffic flow rates for each source-destination pair are estimated by model-free reinforcement learning-based solutions. The experiments are conducted on a server equipped with an Intel(R) Xeon(R) CPU E5-2698 v4 at 2.20GHz, an NVIDIA Tesla V100 GPU, and 512GB of DDR4 RAM.

Baselines: We compare AlphaRoute with the following methods:

1. Minimum free-flow travel latency (minLat): each vehicle selects the route with minimum free-flow travel latency.
2. Minimum travel distance (minDis): each vehicle selects the route with minimum travel distance.
3. Minimum number of traffic lights (minLig): each vehicle selects the route with a minimum number of traffic lights.
4. Approximate equilibrium algorithm for congestion

Algorithm	Beijing Sub-district, Beijing				Upper East Side, Manhattan			
	Alat	Adis	AWai	ADet	Alat	Adis	AWai	ADet
minDis	450.91	6017.30	93.41	0.00	1009.56	1798.48	237.70	0.00
minLat	425.57	6114.80	79.99	97.50	894.32	1814.20	207.28	15.72
minLig	445.34	6023.12	94.23	12.01	994.23	1803.23	243.82	23.70
AECG	400.67	6134.21	65.76	178.21	751.23	1813.21	199.76	23.78
GOR	399.93	6151.97	56.94	134.67	744.60	1818.24	168.88	19.77
AlphaRoute	392.82	6115.13	50.83	97.83	692.53	1806.97	161.70	8.49

Table 2: Performance of AlphaRoute on real-world datasets as compared with several baseline methods.

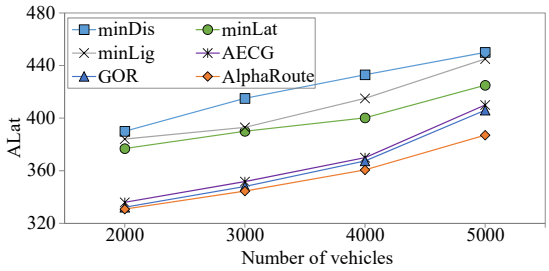


Figure 4: Scalability of AlphaRoute.

games (AECG) (Ravindran Vijayalakshmi and Skopalik 2020): This algorithm generates a sequence of improved moves that converge to an approximate pure Nash equilibrium in a polynomial number of best-response moves. It divides the players into blocks based on their costs, and players of two consecutive blocks are scheduled to make improving moves until converging to the equilibrium.

5. Global Optimal Route (GOR) (Li, Chen, and Shang 2021). This algorithm applies Depth-First Search (DFS) on the road network to find all possible routes from o_i to d_i. Then, a pruning technique is adopted enabling early termination of DFS technique.

Evaluation Metrics: We adopt four metrics to measure the performance of routing choices scheduled by different algorithms, which are introduced as follows:

1. Average travel latency (ALat), i.e., $ALat = \frac{1}{N}\sum_{i\in\mathcal{N}} q_T^i$.
2. Average travel distance (ADis), i.e., $ADis = \frac{1}{N}\sum_{i\in\mathcal{N}}\sum_{e\in p_i} len(e)$.
3. Average waiting time (AWai), i.e., $AWai = \frac{1}{N}\sum_{i\in\mathcal{N}} dur^i$, where dur^i means the duration that the velocity is zero.
4. Average detour distance (ADet), i.e., $ALat = \sum_{i\in\mathcal{N}}(\sum_{e\in p_i} len(e) - \sum_{e\in p_i^*} len(e))$, where p_i^* is the route choices by minDis.

Experiment Results

Overall Analysis. Tab.2 lists the performance of AlphaRoute in terms of ALat, ADis, AWai, and ADet as compared with minLat, minDis, minLig, AECG, and GOR. We have the following observations:

1. Alat is an overall metric for evaluating the performance of coordinated traffic assignment, where a smaller value means less time spent on the road. minLat, minDis and minLig need higher average latency, which means more time is wasted on the road. minLat achieves better performance than minDis, since minDis makes sure minimum distance to the destination while the actual travel time might be higher than expected due to congestion caused by uncoordinated traffic assignments. AlphaRoute achieves the best performance in terms of average latency (Alat), e.g., the improvement is $(450.91 - 392.82)/450.91 = 12.88\%$ as compared with minDis and each driver can save an average of 58 seconds for each journey. The reason behind this improvement comes from two aspects. At first, a region planner generates several strategies under different assignment objectives, where each strategy is a user equilibrium. All generated strategies form the action space. Consequently, the action and state space are significantly decreased, alleviating the curse of dimensionality problem, which is helpful to the searching algorithm. From the other aspect, AlphaRoute can simulate and evaluate the future traffic for sufficient long steps, and apply the simulation results of several steps to make an informed and effective decision. As in GOR, it only considers the decisions of the next step during the search, i.e., it could not take the future traffic evolution into consideration. Consequently, AlphaRoute has the lowest travel latency, achieving the best performance.

2. In terms of travel distance ADis, GOR has the longest travel distance since GOR iterates all possible routes and would assign a vehicle to a farther detour route. More travel distances are caused by the coordinated traffic assignment, i.e., part of vehicles is assigned to a further route to avoid congestion or achieve system optimum. Together with Alat, AlphaRoute achieves a higher travel distance and the lowest travel latency. The average speed of AlphaRoute is the highest, demonstrating the strength in alleviating congestion.

3. In terms of waiting time AWai, AlphaRoute has the least AWai and MinLig has the highest. AWai measures the duration that vehicle velocity is zero, which can be caused by congestion or waiting for traffic lights. AWai means AlphaRoute has the least congestion. The visualization of traffic congestion for different algorithms is presented in our appendix.

4. In terms of detour distance ADet, AECG achieves the highest ADet. The detour distance of a route is compared with the shortest route from the source to the destination.

Therefore, the ADet of minDis is zero. AlphaRoute achieves lower ADet than the other coordinated traffic assignment algorithms, i.e., GOR and AECG, which further demonstrate the strength of AlphaRoute.

Scalability. Fig.4 shows the performance of AlphaRoute under a different number of vehicles N, e.g., when N equals $2000, 3000, 4000$ and 5000, respectively, as compared with the baselines. More vehicles on the road would generate more congestion. When the road is less congested, e.g., when there exist 2000 vehicles, coordinated routing algorithms achieve a smaller ALat than non-coordinated algorithms. The reason behind this phenomenon is partly that non-coordinated algorithms cannot deal with the unbalanced distribution of vehicles, i.e., cannot eliminate congestion. With the number of vehicles N continuing to increase, the strength of AlphaRoute over GOR and AECG is more manifest, since AlphaRoute can be aware of the future traffic evolution and dynamics. It further demonstrates that AlphaRoute is of high scalability.

Ablation Study

In this section, we investigate how different settings affect AlphaRoute.

Number of partitioned regions M. We have divided the road network into the different regions, and the results are shown in Fig.5(a). AlphaRoute achieves better performance with more regions. However, more regions mean larger action and state space, i.e., at each layer, there exist 4^M states. Therefore, to balance the computation complexity and system efficiency, the road network is divided into 4 regions.

Step duration T_{step}. AlphaRoute applies sequence decision-making, and the duration between successive steps is T_{step}. Larger T_{step} means coarser simulations of future traffic states, while smaller T_{step} would lead to heavy computation complexity since the number of states is exponential with the number of steps. We evaluate the performance of AlphaRoute under different T_{step}, as shown in Fig.5(b). Smaller T_{step} leads to higher performance.

Kept states for pruning n_{sta}. In AlphaRoute, it is impossible to evaluate the probability of the actions computed by the local planner. Therefore, the expansion step has to iterate over all possible combinations of actions generated by region planners. A pruning algorithm is applied to early delete states, and make the kept number of states for each layer is at most n_{sta}. The experimental result of AlphaRoute with different n_{sta} is shown in Fig.5(c). Larger n_{sta} leads to better performance. However, larger n_{sta} means higher computation complexity and we choose $n_{sta} = 5$ in our experiments.

Value function. In the pruning step, the value function of each state needs to be evaluated using immediate evaluation function $\Psi(s)$ with minimum efforts. $\Psi(s)$ has to be easily computed and with low complexity, since it needs to be executed frequently. We set $\Psi(s)$ to the following functions, 1) distance: the travel distance to the destination; 2) latency: the free-flow travel latency; 3) balance: the balance rate of vehicle distribution, which is calculated using the variance over traffic saturation on all road segment and 4) Learning method: a graph neural network is applied to approximate the value function. The experiments result of AlphaRoute with different $\Psi(s)$ is shown in Fig.5(d), where $\Psi(s)$ with learning achieves the best performance. The reason is that the former three methods are although easy to compute. They cannot fully approximate the value function. However, the learning method can accurately approximate the value function by using reinforcement learning.

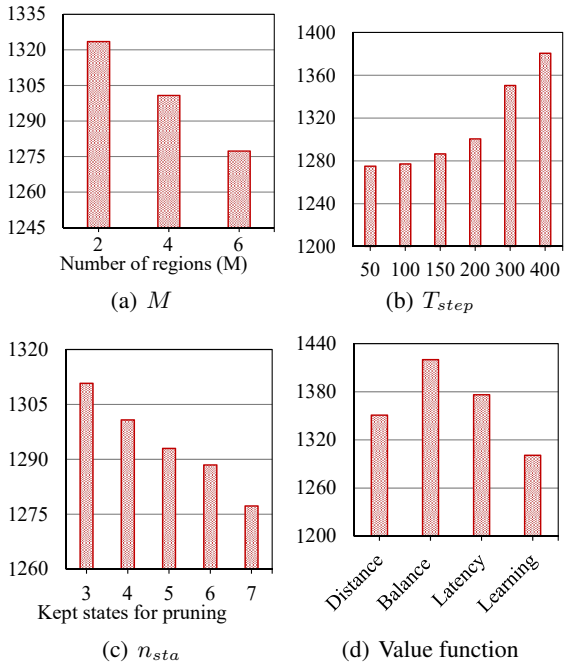


Figure 5: Ablation study, using Alat metric on Manhattan dataset.

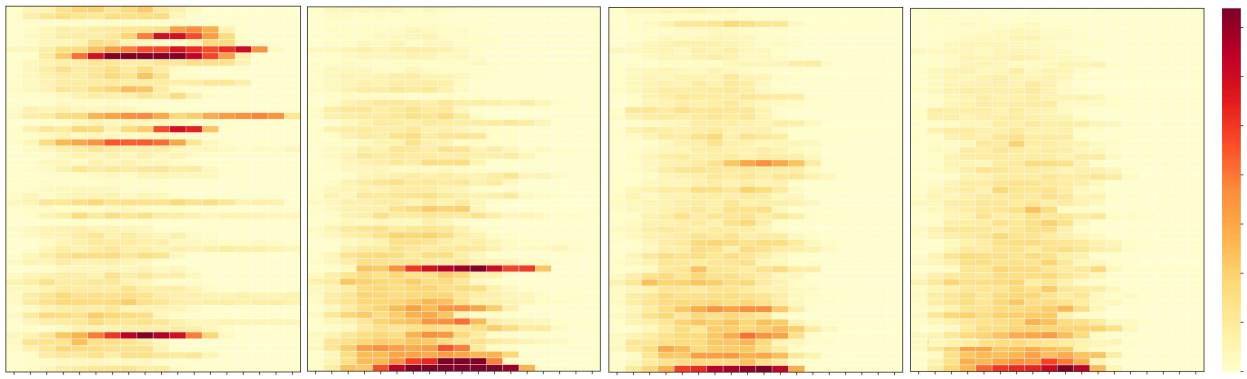
Conclusions

This paper proposes AlphaRoute for coordinating large-scale routes, which is built on value function and Monte Carlo tree search algorithm. AlphaRoute can efficiently and effectively coordinate large-scale routes, and is capable of 1) bridging user fairness and system efficiency, 2) achieving higher search efficiency by alleviating the curse of dimensionality problem, and 3) making an effective and informed route planning by simulating over the future to capture traffic dynamics.

Appendix

A. Proof of Theorem 1

Proof. For objective A, B and C, the cost function $g_i(\boldsymbol{P})$ only correlates with p_i. $\min \sum_{i \in \mathcal{N}^m} g_i(\boldsymbol{P})$ can also be rewritten as $\min_{(p_i \in \mathcal{P}^i)} g_i(p_i)$. For player i, it chooses p_i which minimizes the cost function, i.e., $p_i = argmax_{(p_i \in \mathcal{P}^i)} g_i(p_i)$. If there exists another strategy p_i' that could let i spend lower cost, then, $g_i(p_i') < g_i(p_i)$, which means that p_i is not minimum and is contradictory. Therefore, no player can decrease its cost by unilaterally switching to a different strategy, which would produce one and only



(a) MinDis (b) AECG (c) GOR (d) AlphaRoute

Figure 6: Visualization of congestion for different algorithms, where the horizontal axis means time steps and longitudinal axis is road segment ID. The color bar maps vehicle number to different color and dark red means heavy congestion. AlphaRoute has the least congestion and needs less time (approximately 13 steps) for all vehicles reaching to the destination.

one user equilibrium. As for objective D, the cost function is the travel time which depends on the number of vehicles using the road segments. Under such a scenario, the game is also known as a congestion game. For such a game, the existence and uniqueness of the user equilibrium have been proofed in (Daskin 1985). Therefore, the game has a unique user equilibrium for each defined objective. □

B. Detailed Description of AlphaRoute

AlphaRoute is built on the top of the value function and Monte Carlo tree search algorithm. It consists of five steps, as shown in Fig.7, which are introduced as follows:

a. **Selection:** AlphaRoute begins at the root of the search tree and finishes when the simulation reaches a leaf node. Before the leaf node, an action is selected according to the statistics in the search tree, $a_t = argmax_a(V(s_{t+1}) + u(s_{t+1}))$, with $u(s_t) = c_p \frac{\sqrt{\sum N(s_{t-1})}}{1+N(s_t)}$, where c_p is a constant determining the level of exploration. This search control strategy initially prefers states with equal probability and low visit count, but asymptotically prefers states with high value.

b. **Expansion:** When the visit count $N(s)$ of a state s exceeds a threshold n_{thr}, i.e., $N(s) > n_{thr}$, state s will be expanded. As mentioned earlier, it is impossible to evaluate the probability of the actions computed by the local planner. We fully expand the leaf node s_L to s_{L+1}, i.e., iterate over all possible combinations of actions generated by each region planner.

c. **Pruning:** The expansion step would output an exponential number of states, which can be refined using a state pruning mechanism. For each state, we apply an immediate evaluation $\Psi(\boldsymbol{P})$ to early reject a part of states and keep the number of remaining states at each layer below a threshold n_{sta}. $\Psi(\boldsymbol{P})$ depends on the current positions of all drivers to evaluate the state, without considering the further evolution and dynamics. $\Psi(\boldsymbol{P})$ can be set to 1) positions distribution balance rate; 2) distance to the destination; and 3) free-flow travel latency. Consequently, for the new expanded states $\{s_{L+1}(1), s_{L+1}(2), \cdots, s_{L+1}(k)\}$, the immediate evaluation $\Psi(\boldsymbol{P})$ would obtain the value function $V(s)$ for each state s. Then, using value function to choose the top n_{sta} states, thus making the number of remaining states at each layer below the threshold n_{sta}.

d. **Evaluation:** For a leaf state s_L, we exploit a rollout strategy to evaluate its value $V(s_L)$. Begins with the leaf state, all drivers apply the shortest path routes, i.e., all local planners exploit Objective B for coordinating the route choices. The rollout continues until the end of the game (all drivers have reached their destination). When the game reaches a terminal state, the reward r^i is negatively correlated to the travel time from the leaf state to the end.

e. **Backup:** At the end of the simulation, the rollout statistics are updated in a backward pass through the leaf state to the root state. For each state s_t, $N(s_t) = N(s_t) + 1$, $W(s_t) = W(s_t) + v(s_{t+k}) + kT_{step}$ if state s_t has been traversed, where T_{step} is the duration of each step. The reason behind it is that the reward is travel time, and the difference between two successor states is T_{step}. The overall evaluation of each state is a weighted average of the Monte Carlo estimates, $V(s) = \frac{W(s)}{N(s)}$.

C. More Simulation Results

Visualization of coordinated routes. Fig.6 visualizes the traffic state of road segments at different time steps, where the color means the number of vehicles. According to the color bar, dark red means congestion on the road segment. The following observations can be made according to Fig.6: 1) the congestion area in minDis is largest while that of AlphaRoute is minimum, which demonstrates that AlphaRoute could effective coordinate large-scale routes and alleviates traffic congestion; 2) minDis requires 17 steps for all vehicles reaching their destination, however, AlphaRoute only needs 13 steps. It further demonstrates the strength and efficiency of AlphaRoute.

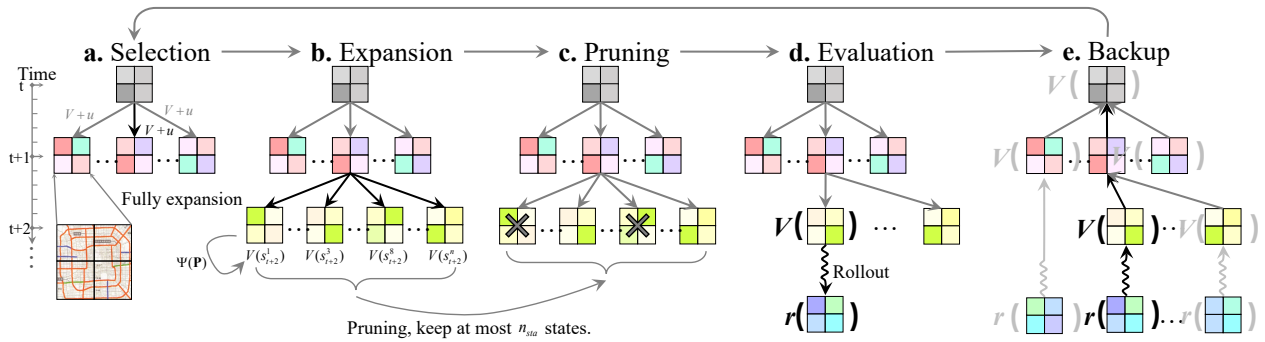


Figure 7: Monte Carlo tree search in AlphaRoute. a. Each simulation traverses the tree by selecting the edge with maximum value $V(s)$, plus a bonus $u(\cdot)$ decaying with repeated visits to encourage exploration. b. The leaf node is fully expanded. c. Each expansion is followed by a pruning mechanism to keep at most n_{sta} states for each layer. d. At the end of a simulation, the leaf node is evaluated by both the value function and a rollout strategy. e. values $V(s)$ are updated to track the mean value of all evaluations and in the subtree below that state.

Acknowledgements

An earlier version of this paper was presented at the 2022 IEEE International Conference on Intelligent Transportation Systems. This paper is supported in part by the National Key Research and Development Program of China under Grant 2021YFB1600402, and 2022YFB4300403, in part by the National Natural Science Foundation of China under Grant 61876023, Grant 62102041, Grant 62203040, and Grant 62272053, and in part by Young Elite Scientists Sponsorship Program by CAST under Grant 2022QNRC001.

References

Anwar, T.; Liu, C.; Vu, H. L.; Islam, M. S.; and Sellis, T. 2018. Capturing the Spatiotemporal Evolution in Road Traffic Networks. *IEEE Transactions on Knowledge and Data Engineering*, 30(8): 1426–1439.

Belov, A.; Mattas, K.; Makridis, M.; Menendez, M.; and Ciuffo, B. 2021. A microsimulation based analysis of the price of anarchy in traffic routing: The enhanced Braess network case. *Journal of Intelligent Transportation Systems*, 1–16.

Chen, L.; Shang, S.; Yao, B.; and Li, J. 2020. Pay your trip for traffic congestion: Dynamic pricing in traffic-aware road networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 582–589.

Çolak, S.; Lima, A.; and González, M. C. 2016. Understanding congested travel in urban areas. *Nature communications*, 7(1): 1–8.

Colini-Baldeschi, R.; Cominetti, R.; Mertikopoulos, P.; and Scarsini, M. 2020. When is selfish routing bad? The price of anarchy in light and heavy traffic. *Operations Research*, 68(2): 411–434.

Correa, J.; Cristi, A.; and Oosterwijk, T. 2021. On the price of anarchy for flows over time. *Mathematics of Operations Research*.

Dai, X.; Zhao, C.; Wang, X.; Lv, Y.; Lin, Y.; and Wang, F.-Y. 2022. Image-based traffic signal control via world models. *Frontiers of Information Technology & Electronic Engineering*, 23(12): 1795–1813.

Daskin, M. S. 1985. Urban transportation networks: Equilibrium analysis with mathematical programming methods. *Transportation Science*, 19(4): 463–466.

Delarue, A.; Anderson, R.; and Tjandraatmadja, C. 2020. Reinforcement learning with combinatorial actions: An application to vehicle routing. *Advances in Neural Information Processing Systems*, 33: 609–620.

Frank, M. 1981. The braess paradox. *Mathematical Programming*, 20(1): 283–302.

Fu, Z.-H.; Qiu, K.-B.; and Zha, H. 2021. Generalize a small pre-trained model to arbitrarily large tsp instances. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 7474–7482.

Gollapudi, S.; Kollias, K.; and Panigrahi, D. 2019. You get what you share: Incentives for a sharing economy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 2004–2011.

Ji, Y.; and Geroliminis, N. 2012. On the spatial partitioning of urban transportation networks. *Transportation Research Part B: Methodological*, 46(10): 1639–1656.

Kang, Y.; Liu, S.; Zhang, H.; Han, Z.; Osher, S.; and Poor, H. V. 2021. Task Selection and Collision-Free Route Planning for Mobile Crowdsensing Using Multi-Population Mean-Field Games. *IEEE Transactions on Green Communications and Networking*, 5(4): 1947–1960.

Klein, I.; Levy, N.; and Ben-Elia, E. 2018. An agent-based model of the emergence of cooperation and a fair and stable system optimum using ATIS on a simple road network. *Transportation research part C: emerging technologies*, 86: 183–201.

Koh, S.; Zhou, B.; Fang, H.; Yang, P.; Yang, Z.; Yang, Q.; Guan, L.; and Ji, Z. 2020. Real-time deep reinforcement learning based vehicle navigation. *Applied Soft Computing*, 96: 106694.

Li, K.; Chen, L.; and Shang, S. 2021. Towards alleviating traffic congestion: optimal route planning for massive-scale

trips. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 3400–3406.

Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*.

Li, Y.; Zheng, Y.; and Yang, Q. 2019. Efficient and effective express via contextual cooperative reinforcement learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 510–519.

Liang, E.; Su, Z.; Fang, C.; and Zhong, R. 2022. OAM: An Option-Action Reinforcement Learning Framework for Universal Multi-Intersection Control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 4550–4558.

Luo, G.; Yuan, Q.; Li, J.; Wang, S.; and Yang, F. 2022a. Artificial Intelligence Powered Mobile Networks: From Cognition to Decision. *IEEE Network*, 36(3): 136–144.

Luo, G.; Yuan, Q.; Zhou, H.; Cheng, N.; Liu, Z.; Yang, F.; and Shen, X. S. 2018. Cooperative vehicular content distribution in edge computing assisted 5G-VANET. *China Communications*, 15(7): 1–17.

Luo, G.; Zhang, H.; Wang, X.; Yuan, Q.; Li, J.; and Wang, F.-Y. 2022b. ACP Based Large-Scale Coordinated Route Planning: From Perspective of Cyber-Physical-Social Systems. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 1842–1849.

Luo, G.; Zhang, H.; Yuan, Q.; Li, J.; and Wang, F.-Y. 2022c. ESTNet: Embedded Spatial-Temporal Network for Modeling Traffic Flow Dynamics. *IEEE Transactions on Intelligent Transportation Systems*, 23(10): 19201–19212.

Luo, G.; Zhou, H.; Cheng, N.; Yuan, Q.; Li, J.; Yang, F.; and Shen, X. 2021. Software-Defined Cooperative Data Sharing in Edge Computing Assisted 5G-VANET. *IEEE Transactions on Mobile Computing*, 20(3): 1212–1229.

Ma, X.; Yu, H.; Wang, Y.; and Wang, Y. 2015. Large-scale transportation network congestion evolution prediction using deep learning theory. *PloS one*, 10(3): e0119044.

Morandi, V. 2021. Bridging the user equilibrium and the system optimum in static traffic assignment: how the cooperation among drivers can solve the congestion problem in city networks. *arXiv preprint arXiv:2105.05804*.

Rahili, S.; Riviere, B.; Olivier, S.; and Chung, S.-J. 2018. Optimal Routing for Autonomous Taxis using Distributed Reinforcement Learning. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, 556–563.

Ravindran Vijayalakshmi, V.; and Skopalik, A. 2020. Improving approximate pure nash equilibria in congestion games. In *International Conference on Web and Internet Economics*, 280–294. Springer.

Shou, Z.; and Di, X. 2020. Multi-agent reinforcement learning for dynamic routing games: A unified paradigm. *arXiv preprint arXiv:2011.10915*.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587): 484–489.

Su, H.; Zhong, Y. D.; Dey, B.; and Chakraborty, A. 2022. EMVLight: A decentralized reinforcement learning framework for efficient passage of emergency vehicles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 4593–4601.

Sun, P.; Li, J.; Guo, Z.; Xu, Y.; Lan, J.; and Hu, Y. 2019. SINET: Enabling Scalable Network Routing with Deep Reinforcement Learning on Partial Nodes. In *Proceedings of the ACM SIGCOMM 2019 Conference Posters and Demos*, SIGCOMM Posters and Demos '19, 88–89. New York, NY, USA: Association for Computing Machinery. ISBN 9781450368865.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, H.; Tang, H.; Hao, J.; Hao, X.; Fu, Y.; and Ma, Y. 2020. Large Scale Deep Reinforcement Learning in Wargames. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 1693–1699. IEEE.

Yu, J. J. Q.; Yu, W.; and Gu, J. 2019. Online Vehicle Routing With Neural Combinatorial Optimization and Deep Reinforcement Learning. *IEEE Transactions on Intelligent Transportation Systems*, 20(10): 3806–3817.

Zhang, H.; Luo, G.; Li, Y.; and Wang, F.-Y. 2022. Parallel Vision for Intelligent Transportation Systems in Metaverse: Challenges, Solutions, and Potential Applications. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 1–14.

Zhang, X.; Liu, W.; and Waller, S. T. 2019. A network traffic assignment model for autonomous vehicles with parking choices. *Computer-Aided Civil and Infrastructure Engineering*, 34(12): 1100–1118.

Zhao, J.; Mao, M.; Zhao, X.; and Zou, J. 2021. A Hybrid of Deep Reinforcement Learning and Local Search for the Vehicle Routing Problems. *IEEE Transactions on Intelligent Transportation Systems*, 22(11): 7208–7218.

Zhou, H.; Xu, W.; Bi, Y.; Chen, J.; Yu, Q.; and Shen, X. S. 2017. Toward 5G spectrum sharing for immersive-experience-driven vehicular communications. *IEEE Wireless Communications*, 24(6): 30–37.

Zhou, Q.; Gu, J.; Lu, X.; Zhuang, F.; Zhao, Y.; Wang, Q.; and Zhang, X. 2021. Modeling heterogeneous relations across multiple modes for potential crowd flow prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 4723–4731.