Real-time Lane Configuration with Coordinated Reinforcement Learning

Udesh Gunarathna(✉), Hairuo Xie, Egemen Tanin, Shanika Karunasekara, and Renata Borovica-Gajic

School of Computing and Information Systems
The University of Melbourne, Victoria, Australia
pgunarathna@student.unimelb.edu.au, {xieh, etanin, karus, renata.borovica}@unimelb.edu.au

Abstract. Changing lane configuration of roads, based on traffic patterns, is a proven solution for improving traffic throughput. Traditional lane-direction configuration solutions assume pre-known traffic patterns, hence are not suitable for real-world applications as they are not able to adapt to changing traffic conditions. We propose a dynamic lane configuration solution for improving traffic flow using a two-layer, multi-agent architecture, named Coordinated Learning-based Lane Allocation (CLLA). At the bottom-layer, a set of reinforcement learning agents find a suitable configuration of lane-directions around individual road intersections. The lane-direction changes proposed by the reinforcement learning agents are then coordinated by the upper level agents to reduce the negative impact of the changes on other parts of the road network. CLLA is the first work that allows city-wide lane configuration while adapting to changing traffic conditions. Our experimental results show that CLLA can reduce the average travel time in congested road networks by 20% compared to an uncoordinated reinforcement learning approach.

Keywords: Reinforcement Learning · Spatial Database · Graphs.

1 Introduction

The goal of traffic optimization is to improve traffic flows in road networks. Traditional solutions normally assume that the structure of road networks is static regardless of how the traffic changes in real-time [6]. A less-common way to optimize traffic is by changing road network configurations at real time. We focus on dynamic lane-direction changes, which can help balance the usage of traffic lanes in many circumstances, e.g. when the traffic lanes in one direction become congested while the traffic lanes in the opposite direction are underused [20, 11].

The impact of dynamic lane-direction configurations can be shown in the following example (Figure 1). In Figure 1a, there are 4 north-bound lanes and 4 south-bound lanes. Traffic is congested in the north-bound lanes. Figure 1b shows the dramatic change of traffic flow after lane-direction changes are applied, where the direction of E, F and G is reversed. The north-bound vehicles are

(a) Traffic before lane-direction change

(b) Traffic after lane-direction change

Fig. 1: The impact of lane-direction change on traffic flow. There are 20 vehicles moving in the north-bound direction and 2 vehicles moving in the south-bound direction.

distributed into the additional lanes, resulting in a higher average speed of the vehicles. At the same time, the number of south-bound lanes is reduced to 1. Due to the low number of south-bound vehicles, the average speed of south-bound traffic is not affected. The lane-direction change helps improve the overall traffic efficiency in this case. There is no existing approach for applying such lane-direction changes at the network level at real-time, which can help improve traffic efficiency of a whole city. We aim to scale this to city-wide areas. The emergence of connected autonomous vehicles (CAVs) [14] can make such large-scale dynamic lane-direction changes a common practice in the future. Compared to human-driven vehicles, CAVs are more capable of responding to a given command in a timely manner [4]. CAVs can also provide detailed traffic telemetry data to a central traffic management system in real time, which is important to dynamic traffic optimization.

In order to optimize the flow of the whole network, one needs to consider the impact of possible lane-direction changes on all the other traffic lanes. In many circumstances, one cannot simply allocate more traffic lanes at a road segment for a specific direction when there is more traffic demand in that direction. This is because a lane-direction change at a road segment can affect not only the flow in both directions at the road segment but also the flow at other road segments. Existing solutions for computing lane-direction configurations [21, 4, 9] do not consider the impact of changes at the network level due the assumption that future traffic dynamics are known beforehand at the beginning of the calculation which is unrealistic for practical applications. More importantly, the computation time can be very high with the existing approaches as they aim to find the optimal configurations based on linear programming, and hence are not suitable for frequent recomputation over large networks.

To address the issues mentioned above: (1) perform in real-time; and (2) having less computational complexity, we propose a multi-agent, scalable, and effective solution, called Coordinated Learning-based Lane Allocation (CLLA), for optimizing lane-directions in dynamic traffic environments. CLLA uses a two-layer

architecture. The bottom layer consists of a set of reinforcement learning agents (*RL Agents*) that operate at the intersection level. A RL Agent finds suitable lane-direction changes for the road segments that connect to a specific intersection. The RL Agents use reinforcement learning [17], which helps determine the best changes based on multiple dynamic factors. The RL Agents send the proposed lane-direction changes to the upper layer, which consists of a set of *Coordinating Agents* who evaluate the global impact of the proposed lane-direction changes and decide what changes should be made to the traffic lanes. The decision is sent back to the RL Agents, which will make the changes accordingly. The main contributions of our work are as follows:

- We formalize a lane-direction optimization problem.
- We propose a first-of-its-kind solution, CLLA, for efficient dynamic optimization of lane-directions that uses reinforcement learning to capture dynamic changes in the traffic.
- Our experiments with real-world data shows that CLLA improves travel time by 20% compared to an uncoordinated RL Agent solution.

2 Related Work

2.1 Learning-based Traffic Optimization

Existing traffic optimization algorithms are commonly based on traffic flow optimization with linear programming [7, 6, 10]. They are suitable computing optimization solutions if traffic demand and congestion levels are relatively static. When there is a significant change in the network, the solutions normally need to be re-computed from scratch. Due to the high computational complexity of finding an optimal solution, these algorithms are not suitable for highly dynamic traffic environments and not suitable for applications where real-time information are used as an input.

With the rise of reinforcement learning [16], a new generation of traffic optimization algorithms have emerged [18, 22, 13]. In reinforcement learning, an agent can find the rules to achieve an objective by repeatedly interacting with an environment. The interactive process can be modelled as a finite Markov Decision Process, which requires a set of states S and a set of actions A per state. Given a state s of the environment, the agent takes an action a. As the result of the action, the environment state may change to s' with a reward r. The agent then decides on the next action in order to maximize the reward in the next round. Reinforcement learning-based approaches can suggest the best actions for traffic optimization given a combination of network states, such as the queue size at intersections [2, 1]. They have an advantage over linear programming-based approaches, since if trained well, they can optimize traffic in a highly dynamic network. In other words, there is no need to re-train the agent when there is a change in the network. For example, Arel et al. show that a multi-agent system can optimize the timing of adaptive traffic lights based on reinforcement learning [1]. Different to the existing approaches, our solution uses

reinforcement learning for optimizing lane-directions which was not considered before.

A common problem with reinforcement learning is that the state space can grow exponentially when the dimensionality of the state space grows linearly. The fast growth of the state space can make reinforcement learning unsuitable for large scale deployments. This problem is known as the *curse of dimensionality* [3]. A common way to mitigate the problem is by using a set of distributed agents that operate at the intersection level. This approach has been used for dynamic traffic signal control [5]. Different to the existing work we use this for dynamic lane-direction configurations.

Coordination of multi-agent reinforcement learning can be achieved through a joint state space or through a coordination graph [8]. Such techniques, however, require agents to be trained on the targeted network. Since our approach uses an implicit mechanism to coordinate (Section 4.3), once an agent is trained, it can be used in any road network.

2.2 Lane-direction Configurations

Research shows that dynamic lane-direction changes can be an effective way to improve traffic efficiency [20]. However, existing approaches for optimizing lane-directions are based on linear programming [4, 21, 9], which are unsuitable for dynamic traffic environments dues to their high computational complexity. For example, Chu et al. uses linear programming to make lane-allocation plans by considering the schedule of connected autonomous vehicles [4]. Their experiments show that the total travel time can be reduced. However, the computational time grows exponentially when the number of vehicles grows linearly, which can make the approach unsuitable for highly dynamic traffic environments. The high computational costs are also inherent to other approaches [21, 9]. Furthermore, all these approaches assume the exact knowledge of traffic demand over the time horizon is known beforehand; this assumption does not hold when traffic demand is stochastic [12]. On the contrary, our proposed approach CLLA is lightweight and can adapt to highly dynamic situations based on reinforcement learning. The reinforcement learning agents can find effective lane-direction changes for individual road intersections even when traffic demand changes dramatically. *To the best of our knowledge, this is the first work for lane-direction allocation by observing real-time traffic information.*

3 Problem Definition

Definition 1. *Road network graph: A road network graph $G_t(V, E)$ is a representation of a road network at time t. Each edge $e \in E$ represents a road segment. Each vertex $v \in V$ represents a start/end point of a road segment.*

Definition 2. *Lane configuration: The lane configuration of an edge e, lc_e, is a tuple with two numbers, each of which is the number of lanes in a specific*

direction on the edge. The sum of the two numbers is always equal to the total number of lanes on the edge.

Definition 3. *Dynamic lane configuration: The dynamic lane configuration of an edge e at time t, $lc_e(t)$, is the lane configuration that is used at the time point.*

Definition 4. *Travel cost: The travel cost of a vehicle i that presents at time t, $TC_i(t)$, is the length of the period between t and the time when the vehicle reaches its destination.*

Definition 5. *Total travel cost: The total travel cost of vehicles that present at time t, TTC(t), is the sum of the travel costs of all the vehicles. That is, $TTC(t) = \sum_{(i=1)}^{n} TC_i(t)$, where n is the number of vehicles.*

PROBLEM STATEMENT. Given a set of vehicles at time t and the road network graph $G_{t-1}(V, E)$ from time $t-1$, find the new graph $G_t(V, E)$ by computing dynamic lane configuration ($lc_e(t)$) for all the edges in E such that the total travel cost $TTC(t)$ is minimized.

4 Coordinated Learning-based Lane Allocation (CLLA)

To solve the optimization problem defined in Section 3, we propose Coordinated Learning-based Lane Allocation (CLLA) solution. CLLA uses a two-layer multi-agent architecture, as shown in Figure 2. The bottom layer consists of a set of RL Agents that are responsible for optimizing the direction of lanes connected to specific intersections. The lane-direction changes that are decided by the RL Agents are aggregated and evaluated by a set of Coordinating Agents at the upper layer, with the aim to resolve conflicts between the RL agents' decisions.

CLLA provides a scalable solution for dynamic lane configuration at the road network level as traffic patterns changes in real-time. CLLA uses reinforcement learning to help optimize lane-direction configurations, which allows optimization in a high variety of real-time traffic conditions. In addition, CLLA achieves coordination between the RL Agents by considering the impact of a potential lane-direction change on different parts of the road network. As detailed later, CLLA only needs to know partial information about vehicle paths in addition to certain real-time traffic conditions, such as intersection queue lengths and lane configuration of road segments, which can be obtained from inductive-loop traffic detectors.

CLLA operates in the following manner. A RL Agent in the bottom layer observes the local traffic condition around a specific intersection. The RL Agents make decisions on lane-direction changes independently. Whenever a RL Agent needs to make a lane-direction change, it sends the proposed change to the Coordinating Agents in the upper layer. The RL Agents also send certain traffic information to the upper layer periodically. The Coordinating Agents evaluate whether a proposed change would be beneficial at the global level based on the received information. The Coordinating Agents may allow or deny a lane-direction

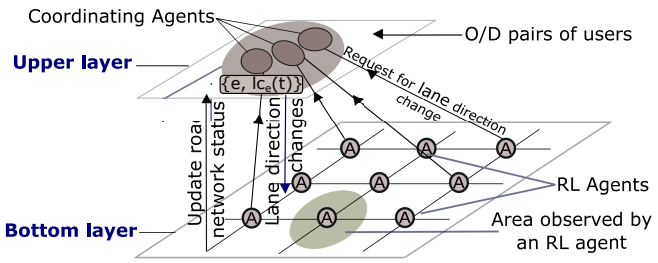


Fig. 2: An overview of the CLLA's architecture

change request. It may also decide to make further changes in addition to the proposed changes. After the evaluation, the Coordinating Agents inform the RL Agents of the changes to be made.

4.1 CLLA Algorithm

Algorithm 1 shows the entire optimization process of CLLA. During one iteration of the algorithm, each RL Agent proposes the lane-direction changes around a specific road intersection using the process detailed in Section 4.2. When it is time to evaluate the proposed changes, the system uses the *Global Impact Evaluation* algorithm (Section 4.3) to quantify the conflicts between the proposed changes and finds coordinated lane-direction changes (Line 8). The coordinated lane-direction changes are then applied to the road segments (Line 10-11).

4.2 Reinforcement Learning Agent (RL Agent)

In CLLA, the RL Agents use Q-learning technique [19] to find suitable lane-direction changes based on real-time traffic conditions. The Q-learning algorithm aims to find a policy that maps a state to an action. The algorithm relies on an *action value function*, $Q(s, a)$, which computes the quality of a state-action combination. Q-learning tries to find the optimal policy that leads to the maximum action value. Q-learning updates the action-value function using an iterative process as shown in Equation 1.

$$Q^{new}(s_t, a_t) = (1 - \alpha).Q(s_t, a_t) + \alpha(r_{t+1} + \gamma.max_a Q(s_{t+1}, a)) \quad (1)$$

where s is the current state, a is a specific action, s_{t+1} is the next state as a result of the action, $max_a Q(s_{t+1}, a)$ is the estimated optimal action value in the next state, value r_{t+1} is an observed reward at the next state, α is a learning rate and γ is a discount factor. In CLLA, the states, actions and rewards used by the RL Agents are defined as follows.

States: A RL Agent can work with four types of states as shown below.

– The first state represents the current traffic signal phase at an intersection.

Algorithm 1: Coordinated Learning Lane Allocation (CLLA)

Input: t_a, time between two coordinating operations
Input: LLC, set of edge-change pairs proposed by the RL Agents
Input: G, Road Network
Input: CLC, set of edge-change pairs given by the Coordinating Agents
$t \leftarrow 0$, $t_{step} \leftarrow 0$
while *True* **do**
  foreach *agent* $\in$ *RL Agents* **do**
    determine the best lane-direction change for all the edges (road segments) that connect to the vertex $v \in G$ (intersection) controlled by the *agent*
    foreach *edge e that needs a lane-direction change* **do**
      $LLC.insert(\{e, lc_e(t)\})$
  if $t_a = t_{step}$ **then**
    $CLC \leftarrow$ **Global Impact Evaluation(**LLC**)**
    $LLC \leftarrow \emptyset$, $t_{step} \leftarrow 0$
    foreach $\{e, lc_e(t)\}$ *in* CLC **do**
      apply the lane-direction change to e
    $t \leftarrow t + 1$
  $t_{step} \leftarrow t_{step} + 1$

- The second state represents the queue length of incoming vehicles that are going to pass the intersection without turning.
- The third state represents the queue length of incoming vehicles that are going to turn at the intersection.
- The fourth state represents the queue length of outgoing vehicles, i.e., the vehicles that have passed the intersection.

Although it is possible to add other types of states, we find that the combination of the four states can work well because the combination of four states provides; i) information about both incoming and outgoing traffic, ii) from which road to which road vehicles are waiting to move, iii) current traffic signal information.

Actions: We denote the two directions of a road segment as *upstream* and *downstream*. There are three possible actions: increasing the number of upstream lanes by 1, increasing the number of downstream lanes by 1 or keeping the current configuration. When the number of lanes in one direction is increased, the number of lanes in the opposite direction is decreased at the same time. Since a RL Agent controls a specific road intersection, the RL Agent determines the action for each individual road segment connected to the intersection.

We introduced an action restriction mechanism in RL Agents. Changing lane-direction of a road segment takes time as existing vehicles on that road segment should move out before reversing the lane-direction. Therefore, it takes an even longer time to recover from an incorrect lane-direction decision taken by a RL

Agent while learning. In order to stabilize the learning, a RL Agent is allowed to take a lane-changing action only when there is a considerable difference between upstream and downstream traffic. The use of this restriction also provides a way to resolve conflicting actions between neighboring RL Agents. When two RL Agents connected to the same road segment want to increase the number of lanes in different directions, the priority is given to the action, which allocates more lanes to the direction with a higher traffic volume.

Rewards: We define the rewards based on two factors. The first factor is the waiting time of vehicles at an intersection. When the waiting time decreases, there is generally an improvement of traffic efficiency. Hence the rewards should consider the difference between the current waiting time and the updated waiting time of all the vehicles that are approaching the intersection. The second factor is the difference between the length of vehicle queues at different approaches to an intersection. When the queue length of one approaching road is significantly longer than the queue length of another approaching road, there is a higher chance that the traffic becomes congested in the former road. Therefore we need to penalize the actions that increase the difference between the longest queue length and the shortest queue length. The following reward function combines the two factors. A parameter β is used to give weights for the two factors. We normalized the two factors to stabilize the learning process by limiting reward function between 1 to -1. To give equal priority to both factors, we set β to 0.5 in the experiments.

$$R = (1-\beta) \times \frac{Current\_wait\_time - Next\_wait\_time}{\max(Next\_wait\_time, Current\_wait\_time)}$$

$$-\beta \times \frac{Queue\_length\_difference}{Aggregated\_road\_capacity}$$

4.3 Coordinating Agent

Given a locally optimized lane-direction change, Coordinating Agents check whether the change can help improve traffic efficiency in surrounding areas based on the predicted traffic demand and the current traffic conditions. If a proposed change is beneficial, it can be actioned. Otherwise, it is not allowed by CLLA.

We first, explain the process of coordinating lane-direction changes using a simple example shown in Figure 3, where two vehicles are moving from left to right while four other vehicles are moving in the opposite direction. Let us assume that the RL Agent for road segment e_1 proposes to increase the number of lanes from A to B because there is no vehicle in the opposite direction on e1 now. Although such a lane-direction change would help reduce the travel time on e_1, it may conflict with the predicted traffic demand on e_2. The reason is that four vehicles will go through e_2 from right to left (from C to B) but only two vehicles will go through the same road segment from left to right (from B to C). Therefore, the overall traffic demand on e2 will be from right to left (from C to B).

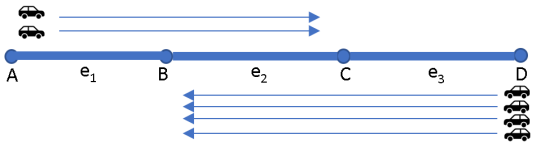


Fig. 3: The vehicles on a road with three road links, e_1, e_2 and e_3. The vehicles will follow the paths shown in arrows.

However, by increasing the number of lanes from left to right on e_1, the number of lanes in the opposite direction decreases, which is likely to cause a drop of traffic flow speed from B to A. The traffic congestion can eventually propagate to the road segment from C to B. This is not ideal as the overall traffic demand would be from C to B. Consequently, increasing the number of lanes from left to right (from A to B) on e_1 is not beneficial and should not be actioned.

Due to the dynamic nature of traffic, the Coordinating Agents may not need to consider the full path of vehicles when evaluating the proposed changes based on the predicted traffic demand. This is because the route of vehicles may change dynamically at real time, especially in the era of connected autonomous vehicles when traffic optimization can be performed frequently. Instead of collecting the full path of vehicles, the Coordinating Agents can collect the path within a *lookup distance*. For example, assuming the lookup distance is 200 metres, the Coordinating Agents only need to know the road segments that the vehicles will pass within the next 200 metres from their current locations.

When there is no conflict between a proposed lane-direction change and the predicted traffic demand, CLLA evaluates the benefit of the proposed change based on the current traffic conditions. Our implementation considers one specific type of traffic condition, the current queue length at road junctions. If a lane-direction change can lead to a lower traffic speed on a road segment, which has a longer queue than the road segment in the opposite direction, the lane-direction change is not allowed. This is because a lower traffic speed can lead to an even longer queue, which can decrease traffic efficiency.

The coordination of lane-direction changes is performed at a certain interval. The time between two coordinating operations is the *assignment interval*, within which the proposed lane-direction changes are actioned, the predicted traffic demand and the current traffic condition are aggregated at the Coordinating Agents.

Global Impact Evaluation Algorithm: The Coordinating Agents use Global Impact Evaluation Algorithm (Algorithm 2) to quantify the conflicts between lane-direction changes. The algorithm takes lane-direction changes that are proposed by the RL Agents as an input (LLC). The input consists of the road and the lane-direction change (lc) proposed by each RL Agent. First, the algorithm finds the neighboring road segments affected by all the changes proposed by the RL Agents (Line 3). For each neighboring road segment, the algorithm finds the predicted traffic flow caused by the proposed lane-direction changes

Algorithm 2: Global Impact Evaluation (GIE)

Input: LLC, a set of local lane-direction changes (road id, action pair) proposed by the RL Agents
Input: t, current time
Output: CLC, a set of (road id, action pair) given by the Coordinating Agents

1 $q \leftarrow \emptyset$; $CLC \leftarrow \emptyset$
2 **foreach** $(r, lc_r(t)) \in LLC$ **do**
3 $roads \leftarrow$ Neighboring road segments affected by the lane-configuration $(lc_r(t))$ in r, which are within the lookup distance
4 **foreach** $r_{new} \in roads$ **do**
5 Calculate the predicted traffic flow change in r_{new} due to $lc_r(t)$
6 **if** r_{new} *not in* q **then**
7 $q.add(r_{new})$
8 **foreach** $r_{new} \in q$ **do**
9 $lc_{r_{new}}(t) \leftarrow$ decide the lane-configuration for r_{new} based on predicted traffic
10 **if** $lc_{r_{new}}(t)$ *contains a lane direction change* **then**
11 $CLC.add([r_{new}, lc_{r_{new}}(t)$
12 **if** r_{new} *cannot accommodate predicted traffic flows* **then**
13 mark corresponding change in LLC as a conflict
14 **foreach** $r, lc_r(t) \in LLC$ **do**
15 **if** *no conflicts for* r **then**
16 $CLC.add([r, lc_r(t)])$

(Line 5). Then the algorithm adds affected neighboring road segments to a queue (Line 7).

In the next step, the algorithm visits each road segment in the queue and determines the appropriate lane-direction configuration ($lc_{r_{new}}(t)$) and the conflicts, where a road segment cannot accommodate the predicted traffic flow (Line 9-13). If a lane-direction change needs to be made, for road segment r_{new}, the road segment is added to coordinated lane changes (CLC) (Line 11). If there is a conflict at road segment r_{new} , corresponding lane-direction change proposed by the RL Agents is marked as a conflict (Line 13).

In the last step, the algorithm adds lane-direction changes proposed by the RL Agents to coordinated lane changes if there is no conflict (Line 14-16).

Complexity of Coordinating Process. Let us use m to denote the number of requests from the RL Agents. The complexity of visiting the relevant road segments is $\mathcal{O}(m \times neb)$ where neb is the number of neighboring road segments that connect to a road segment at a road junction. Since the number of road segments connecting with the same junction is normally a small value, neb can be seen as a constant value with a given lookup distance (l_up). Hence the algorithm complexity can be simplified to $\mathcal{O}(m)$. In the worst case, there is a lane-change request for each road segment of $G(V, E)$, leading to a complexity of $\mathcal{O}(|E|)$.

Distributed Version. Since the execution of Global Impact Evaluation algorithm is independent of the order of requests coming from the RL Agents, requests can be processed in a distributed manner using multiple Coordinating Agents. Every Coordinating Agent traverses first depth neighbors and informs changes to other Coordinating Agents. In such a setting, the complexity of the algorithm is $\mathcal{O}(1)$ with $|E|$ number of Coordinating Agents. In this work, we implemented the centralized version (with one Coordinating Agent); however, when applied to very large road networks, the distributed version can be implemented.

5 Experimental Methodology

We compare the proposed algorithm, CLLA, against three baseline algorithms using traffic simulations. We evaluate the performance of the algorithms using synthetic traffic data and real traffic data. We use SMARTS (Scalable Microscopic Adaptive Road Traffic Simulator) [15], a microscopic simulator capable of changing the travelling directions of lanes, for our experiments.

Datasets. The real traffic data contains the taxi trip records from New York City [1]. The data includes the source, the destination and the start time of the taxi trips in the city. We pick an area of Manhattan for simulation (Figure 4) because the area contains a larger amount of taxi trip records than other areas. The road network of the simulation areas is loaded from OpenStreetMap [2]. For a specific taxi trip, the source and the destination are mapped to the nearest OpenStreetMap nodes. The shortest path between the source and the destination is calculated. The simulated vehicles follow the shortest paths generated from the taxi trip data.

We also use a synthetic 7x7 grid network to evaluate how our algorithm performs in specific traffic conditions.

We simulate four traffic patterns with the synthetic road network. A traffic pattern refers to generating vehicles to follow a specific path between a source node and a destination node in the road network.

- **Rush hour traffic (RH)**: In this setup, traffic is generated so that traffic demand is directionally imbalanced to represent rush hour traffic patterns.
- **Bottleneck traffic (BN)**: This setup generates high volume of traffic at the centre of the grid network. This type of traffic patterns create bottleneck links at the center of the network.
- **Mixed traffic (MX)**: Mixed traffic contains both **Rush hour traffic** and **Bottleneck traffic** conditions in the same network.
- **Random traffic (RD)**: Traffic is generated randomly during regular time intervals. Demand changes over time intervals.

Comparison baselines. Different to the proposed solution, CLLA, the existing approaches assume future traffic dynamics are known, hence not practical

[1] https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[2] https://www.openstreetmap.org

Fig. 4: The road network of Midtown Manhattan (MM)

in real-world applications. Due to the lack of comparable solutions, we define three baseline solutions, which are used to compare against CLLA. In our experiments, the traffic signals use static timing and phasing in all solutions. We conduct comparative tests against the following solutions:

- **No Lane-direction Allocations** (**no-LA**): This solution does not do any lane-direction change. The traffic is controlled by static traffic signals only.
- **Demand-based Lane Allocations** (**DLA**): This solution assumes that the full knowledge of estimated traffic demand and associated paths are known at a given time step. DLA computes traffic flow for every edge for both directions by projecting the traffic demand to each associated path. Then it allocates more lanes for a specific direction when the average traffic demand per lane in the direction is higher than the average traffic demand per lane in the opposite direction. Same as CLLA, DLA configures lane-directions at a certain interval, t_a, which is called assignment interval.
- **Local Lane-direction Allocations** (**LLA**): This solution uses multiple learning agents to decide lane-direction changes. The optimization is performed using the approach described in Section 4.2. LLA is similar to CLLA but there is no coordination between the agents.

5.1 Evaluation Metrics

We measure the performance of the solutions based on the following metrics.

Deviation from free-flow travel time: The free-flow travel time of a vehicle is the shortest possible travel time, achieved when the vehicle travels at the speed limit of the roads without slowing down at traffic lights during its entire trip. Deviation from Free-Flow travel Time ($DFFT$) is defined as in Equation 2, where t_a is the actual time and t_f is the free-flow travel time. The lowest value of DFFT is 1, which is also the best value that a vehicle can achieve.

$$DFFT = t_a/t_f \tag{2}$$

Average travel time: The travel time of a vehicle is the duration that the vehicle spends on travelling from its source to its destination. We compute the

Parameter	**Range**	**Default value**
Lookup distance in CLLA	1 - 7	5
Assignment interval in CLLA/DLA (minutes)	0.5 - 3	1

Table 1: Parameter settings

average travel time based on all the vehicles that complete their trips during a simulation. A higher average travel time indicates that the traffic is more congested during the simulation. To make the value robust for network size we present results by subtracting free flow travel time from actual travel time. Our proposed solutions aim to reduce the average travel time.

5.2 Parameter Settings

For LLA and CLLA, the learning rate α is 0.001 and the discount factor used by Q-learning is 0.75. The RL agents are pre-trained, based on the traffic at a single intersection before deployed to all the intersections in a road network. For other parameters of the solutions, we use the default values as shown in Table 1.

6 Experimental Results

6.1 Comparative Tests

Average travel time: Table 2 shows results with synthetic data. As shown in the results, LLA algorithm performs well in rush hour traffic conditions (**RH**). However, it performs poorly when there are bottleneck traffic links (**BN**). This trend is also observed with DLA. When traffic pattern changes frequently (as in **RD**), DLA is not able to estimate the demand hence perform poorly. In contrast, CLLA algorithm performs well in all traffic conditions.

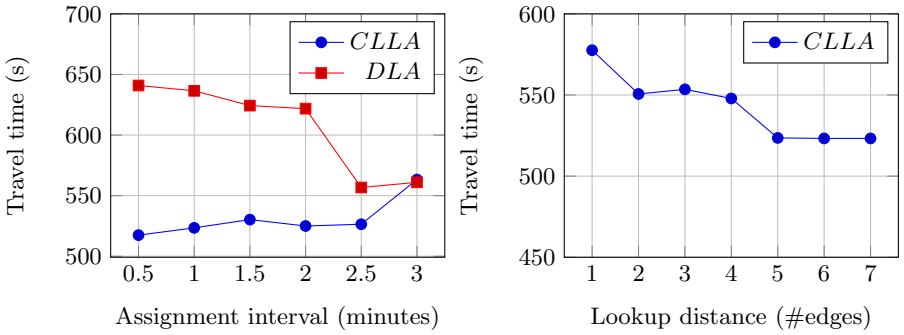
CLLA algorithm outperforms all other baselines in the Manhattan network, as shown in Table 3. CLLA achieves 5% travel time improvement compared to the next best baseline. In traffic engineering terms, this is a significant improvement. The improvement compared to LLA algorithm is around 20%, which highlights the importance of the coordination between RL Agents.

Baseline	Travel Time(s)				% of Vehicles with DFFT>6			
	RH	BN	MX	RD	RH	BN	MX	RD
no-LA	681.08	427.16	506.28	539.89	49.0	4.8	27.7	4.85
LLA	575.59	540.62	561.11	577.6	32.3	24.35	30.5	8.41
DLA	568.02	504.70	493.13	636.51	30.2	16.5	15.5	20.0
CLLA	**568.01**	**428.28**	**449.26**	**523.42**	**32.4**	**5.7**	**14.3**	**3.67**

Table 2: Performance of baselines evaluated using four traffic patterns of the synthetic grid network. **RH, BN, MX, RD** refers to the four synthetic traffic patterns

Baseline	Travel Time(s)	% of Vehicles with DFFT>6
no-LA	604.32	45.9
LLA	585.83	48.6
DLA	496.12	50.7
CLLA	**471.28**	**45.87**

Table 3: Performance of baselines evaluated using New York taxi data



(a) Travel time improvement vs assignment interval

(b) Travel time improvement as lookup distance increases

Fig. 5: Sensitivity analysis with assignment interval and lookup distance

Deviation from free-flow travel time (DFFT): Table 2 and Table 3 show the percentage of vehicles whose travel time is 6 times or more than their free-flow travel time. The results show that CLLA is able to achieve a lower deviation from the free-flow travel time compared to DLA and LLA.

6.2 Sensitivity Analysis

When the assignment interval t_a of DLA increases, travel time decrease, because it is more likely to get a good estimation of traffic demand when the assignment interval is larger, which can lead to more effective optimizations (Figure 5a). Different to DLA, the travel time achieved with CLLA grows slowly with the increase of t_a but it is significantly lower than DLA in most cases. The relatively steady performance of CLLA shows that the coordination between lane-direction changes can help mitigate traffic congestion for a certain period of time in the future. If minimizing the average travel time is of priority, one can set t_a to a very low value, e.g., 0.5 minutes based on the results.

Figure 5b shows that a larger lookup distance can result in a lower average travel time. When the lookup distance increases, CLLA considers more road segments in a vehicle path. This helps identify the conflicting lane-direction changes on the path. Reduction in the average travel time becomes less significant when the lookup distance is higher than 5. This is because the impact of a lane-direction change reduces when the change is further away.

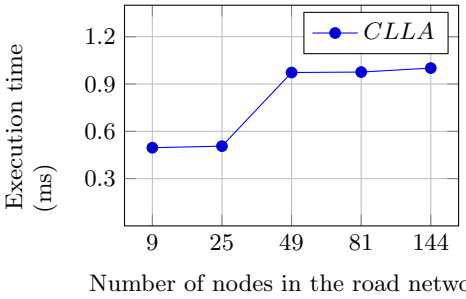


Fig. 6: Execution time for one iteration of GIE algorithm with the road network size. (Lookup distance used = 5)

Figure 6 shows the average execution time of **Global Impact Evaluation** algorithm for one iteration as network size grows. For this test, we build synthetic grid-based road networks. In networks with 9 to 25 nodes, the number of road links on vehicle paths is usually less than the default lookup distance (5). When the number of nodes in a road network is 49, 81 or 144, the number of road links on vehicle paths can be higher than the lookup distance. This is the reason for the increase in execution time when the number of nodes increases from 25 to 49. When the number of nodes is higher than 49, execution is nearly constant, showing that the computation cost does not increase with network size when the lookup distance is fixed.

7 Conclusion

We have shown that effective traffic optimization can be achieved with dynamic lane-direction configurations. Our proposed hierarchical multi-agent solution, CLLA, can help to reduce travel time by combining machine learning and the global coordination of lane-direction changes. The proposed solution adapts to significant changes of traffic demand in a timely manner, making it a viable choice for realizing the potential of connected autonomous vehicles in traffic optimization. Compared to state-of-the-art solutions based on lane-direction configuration, CLLA runs more efficiently, and is scalable to large networks.

An interesting extension would be to incorporate dynamic traffic signals into the optimization process to this work. It would also be interesting to develop solutions that can dynamically change vehicle routes in addition to the lane-direction changes. The dynamic change of speed limit of roads can also be included in an extension to CLLA. Moreover, it is worthwhile to explore how to jointly optimize route allocation and lane directions to improve traffic further.

References

1. Arel, I., Liu, C., Urbanik, T., Kohls, A.: Reinforcement learning-based multi-agent system for network traffic signal control. IET Intelligent Transport Systems **4**(2),

128–135 (2010)
2. Aslani, M., Seipel, S., Mesgari, M.S., Wiering, M.: Traffic signal optimization through discrete and continuous reinforcement learning with robustness analysis in downtown Tehran. Advanced Engineering Informatics **38**, 639 – 655 (2018)
3. Boutilier, C.: Planning, learning and coordination in multiagent decision processes. In: Theoretical Aspects of Rationality and Knowledge. pp. 195–210 (1996)
4. Chu, K.F., Lam, A.Y.S., Li, V.O.K.: Dynamic lane reversal routing and scheduling for connected autonomous vehicles. In: International Smart Cities Conference. pp. 1–6 (2017)
5. El-Tantawy, S., Abdulhai, B.: Multi-agent reinforcement learning for integrated network of adaptive traffic signal controllers. ITSC **14**(3), 1140–1150 (2012)
6. Fleischer, L., Skutella, M.: Quickest flows over time. SIAM J. Comput. **36**(6), 1600–1630 (2007)
7. Ford, L.R., Fulkerson, D.R.: Constructing maximal dynamic flows from static flows. Oper. Res. **6**(3), 419–433 (1958)
8. Guestrin, C., Lagoudakis, M.G., Parr, R.: Coordinated reinforcement learning. In: International Conference on Machine Learning. p. 227–234 (2002)
9. Hausknecht, M., Au, T., Stone, P., Fajardo, D., Waller, T.: Dynamic lane reversal in traffic management. In: ITSC. pp. 1929–1934 (2011)
10. Köhler, E., Möhring, R.H., Skutella, M.: Traffic Networks and Flows over Time, pp. 166–196 (2009)
11. Lambert, L., Wolshon, B.: Characterization and comparison of traffic flow on reversible roadways. Journal of Advanced Transportation **44**(2), 113–122 (2010)
12. Levin, M.W., Boyles, S.D.: A cell transmission model for dynamic lane reversal with autonomous vehicles. Transportation Research Part C: Emerging Technologies **68**, 126 – 143 (2016)
13. Mannion, P., Duggan, J., Howley, E.: An experimental review of reinforcement learning algorithms for adaptive traffic signal control, pp. 47–66 (2016)
14. Narla, S.R.: The evolution of connected vehicle technology: From smart drivers to smart cars to... self-driving cars. ITE Journal **83**, 22–26 (2013)
15. Ramamohanarao, K., Xie, H., Kulik, L., Karunasekera, S., Tanin, E., Zhang, R., Khunayn, E.B.: Smarts: Scalable microscopic adaptive road traffic simulator. ACM Trans. Intell. Syst. Technol. **8**(2) (2016)
16. Ravishankar, N.R., Vijayakumar, M.V.: Reinforcement learning algorithms: Survey and classification. Indian Journal of Science and Technology **10**(1), 1–8 (2017)
17. Sutton, R.S., Barto, A.G.: Introduction to Reinforcement Learning, vol. 135. MIT Press, 1st edn. (1998)
18. Walraven, E., Spaan, M.T., Bakker, B.: Traffic flow optimization: A reinforcement learning approach. Engineering Applications of Artificial Intelligence **52**, 203 – 212 (2016)
19. Watkins, C.J., Dayan, P.: Technical note: Q-learning. Machine learning **8**, 279–292 (1992)
20. Wolshon, B., Lambert, L.: Planning and operational practices for reversible roadways. ITE Journal **76**, 38–43 (2006)
21. Wu, J.J., Sun, H.J., Gao, Z.Y., Zhang, H.Z.: Reversible lane-based traffic network optimization with an advanced traveller information system. Engineering Optimization **41**(1), 87–97 (2009)
22. Yau, K.L.A., Qadir, J., Khoo, H.L., Ling, M.H., Komisarczuk, P.: A survey on reinforcement learning models and algorithms for traffic signal control. ACM Computing Surveys (CSUR) **50**(3) (2017)