Reducing Bus Bunching with Asynchronous Multi-Agent Reinforcement Learning

Jiawei Wang and **Lijun Sun**[*]
McGill University, Montreal, Canada
jiawei.wang4@mail.mcgill.ca, lijun.sun@mcgill.ca

Abstract

The bus system is a critical component of sustainable urban transportation. However, due to the significant uncertainties in passenger demand and traffic conditions, bus operation is unstable in nature and bus bunching has become a common phenomenon that undermines the reliability and efficiency of bus services. Despite recent advances in multi-agent reinforcement learning (MARL) on traffic control, little research has focused on bus fleet control due to the tricky asynchronous characteristic—control actions only happen when a bus arrives at a bus stop and thus agents do not act simultaneously. In this study, we formulate route-level bus fleet control as an asynchronous multi-agent reinforcement learning (ASMR) problem and extend the classical actor-critic architecture to handle the asynchronous issue. Specifically, we design a novel critic network to effectively approximate the marginal contribution for other agents, in which graph attention neural network is used to conduct inductive learning for policy evaluation. The critic structure also helps the ego agent optimize its policy more efficiently. We evaluate the proposed framework on real-world bus services and actual passenger demand derived from smart card data. Our results show that the proposed model outperforms both traditional headway-based control methods and existing MARL methods.

1 Introduction

Public transport has been considered the most critical component in sustainable urban transportation. Providing reliable and efficient services to meet the increasing transit demand has become a major challenge faced by public transit operators. However, in real-world operations, the great uncertainties in traffic conditions (e.g., congestion, incident, and signal control) and dwell time often make bus services unstable [Daganzo and Ouyang, 2019]. Bus bunching has become a common phenomenon in the operation of high-frequency bus services. For a bus service, vehicles essentially leave the departure terminal with a regular frequency/headway (e.g., every 10 min) based on the timetable. If a bus is slightly delayed on the route, it will likely encounter more waiting passengers at the next bus stop, and then the dwell time serving boarding/alighting passengers will also increase. As a result, the delayed bus will be further delayed, and the following bus will become faster due to fewer waiting passengers and shorter dwell time; eventually, two or even more buses will bunch together and travel as a group (e.g., see Fig. 1). Bus bunching has two major negative effects on rider experience: on the one hand, passengers will suffer from long waiting times and then see a group of vehicles arriving together; on the other hand, bus bunching also leads to imbalanced occupancy, since the leading bus in a group will take more passengers and the others will take less. Overall, bus bunching is an urgent issue that undermines the reliability and efficiency of transit services. In general, the severity of bus bunching increases with the service length, and downstream passengers often suffer more than upstream passengers.

Bus holding control is one of the most effective methods to reduce bus bunching. Recently advances in model-free reinforcement learning (RL) have shed new light on the sequential control problem, allowing us to develop efficient and effective control policies for a large system without building an explicit model. In particular, multi-agent reinforcement learning (MARL) provides a learning scheme to explore collaborative policy, which has been shown to be superior in a multi-agent system. The application of MARL in intelligent transportation systems (ITS) has attracted considerable atten-

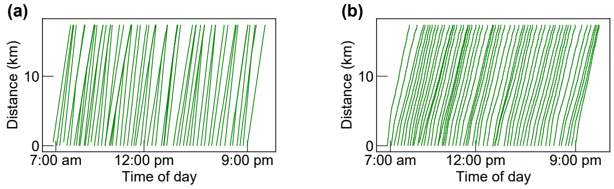

Figure 1: Visualization of bus trajectories with heavy/reduced bus bunching in one day. Each line shows the trajectory of a bus. Panel (a) shows the trajectories over a day without any control/interventions, and panel (b) shows the results of our proposed CAAC framework. The negative effects of bus bunching are more severe on long truck services than on short feeder services.

[*] Contact Author

tion in recent years, with a particular focus on traffic signal control (see e.g., [Chen *et al.*, 2020]). Bus fleet operation is also a unique ITS application where MARL fits very well. On this track, Chen *et al.* [2016] and Alesiani and Gkiotsalitis [2018] proposed MARL models to develop holding control strategies to match a pre-specified headway; however, such a setting is often too restrictive to adapt to the great uncertainties in bus fleet operation. Instead of using a fixed headway, Wang and Sun [2020] proposed a reward function to promote headway equalization in the MARL model. Nevertheless, the models above overlook the fact that buses are not being controlled simultaneously since holding happens only when a bus arrives at a bus stop. In such an asynchronous setting, the default MARL cannot effectively take into account the impact from other agents. To deal with such asynchronicity, Menda *et al.* [2018] proposed an event-driven MARL by designing a reward function associated with macro-action; however, this approach still cannot distinguish the contributions from ego agent and other agents.

To better address the asynchronous issue in bus fleet control, in this paper we propose a new credit assignment framework for asynchronous control (CAAC) by integrating actions from other agents using inductive graph learning. Our main contribution is threefold:

- We establish an asynchronous multi-agent reinforcement learning (ASMR) framework to optimize holding control policy on a bus route to reduce bus bunching.
- To the best of our knowledge, this work is the first to consider the marginal contribution from uncertain events in MARL. This proposed CAAC also has potential to be generalized to other asynchronous control problems.
- We build a learning framework for bus operation in an asynchronous setting based on real-world transit operation data, and evaluate the proposed CAAC with extensive experiments.

2 Related Work

2.1 Holding Control to Avoid Bus Bunching

Bus holding control is a longstanding research topic in transit operation [Daganzo and Ouyang, 2019]. Traditional methods mainly focus on maintaining headway consistency by deriving suitable policies using optimization-based models. However, in practice, this approach has two major limitations. On the one hand, due to the lack of field data, scenarios and optimization models are often over-simplified in many existing studies (e.g., [Daganzo, 2009; Wang and Sun, 2020]). As a result, these models can hardly reproduce reality and scale to large real-world scenarios. On the other hand, these optimization-based models are essentially scenario-specific [Seman *et al.*, 2019], and in practice they require careful design and training for each case. Therefore, these optimization-based models often fail to generalize the policy learned from the past to future and unseen scenarios.

2.2 Asynchronous Multi-agent Control

In this paper, we tackle the asynchronous multi-agent control problem by introducing a new credit assignment scheme. Credit assignment aims to better estimate each agents' own contribution to the team's success for better policy update, and it has been a central research topic in designing reliable MARL algorithms [Chang *et al.*, 2004]. For example, Foerster *et al.* [2018] proposed COMA—a multi-agent actor-critic algorithm with a counterfactual baseline to quantify the contribution from each agent. However, COMA only works for applications with a discrete action space. Recent studies have proposed credit assignments by designing different value decomposition schemes, which can accommodate both discrete and continuous action spaces. For example, Sunehag *et al.* [2018] proposed Value-Decomposition Network (VDN) to measure the impact of each agent on the observed joint reward. Rashid *et al.* [2018] developed QMIX, which improves VDN by adding restrictions on the relation between local and global Q-value. Furthermore, Son *et al.* [2019] developed QTran as a more generalized factorization scheme. These studies have made outstanding contributions to the general credit assignment problem. However, a major limitation is that they only consider local observation-action, which hinders the exploration of more effective cooperation policy. Notably, Wang *et al.* [2020] derived a similar credit assignment framework with Shapely Q-value to better quantify local contributions; however, the asynchronous issue with a varying number of activated agents is still overlooked. Lee *et al.* [2020] studied sequential decision problems involving multiple action variables whose control frequencies are different; still, the framework only considers a pre-specified frequency for each single agent.

3 Holding Control as an ASMR Task

In this paper, we define the forward headway for a specific bus at a certain time as the time duration for it to reach the current location of the preceding bus. Correspondingly, we define the backward headway as the forward headway of its follower. Holding control is a widely used strategy to maintain headway consistency and avoid bus bunching in daily operation [Wang and Sun, 2020]. The key idea of holding control is to let a bus stay longer (i.e., by adding a slack period) at stops in addition to the dwell time for passengers to board/alight. For instance, when a bus gets too close to the preceding vehicle, it can equalize/balance the forward headway and backward headway by holding at a bus stop. In this study, we model bus holding as a control policy in MARL with the goal of improving overall operation efficiency.

Fig. 2 shows the vehicle holding control framework on a bus route, in which we take all the buses running in the system/environment as agents. Note that we label all buses in a sequential order, where bus b_{i+1} follows bus b_i. Bus holding control for a fleet $\{b_i\}$ is a typical application of asynchronous multi-agent control, as holding control is applied only when a bus arrives at a bus stop. We define the basic elements in this framework as follows:

State: Each agent has a local observation of the system. For bus b_i arriving at stop k_j at time t, we denote the state of b_i by $s_{i,t}$, which includes the number of passengers onboard, the forward headway h_i and the backward headway h_{i+1}.

Action: Following Wang and Sun [2020], we model holding

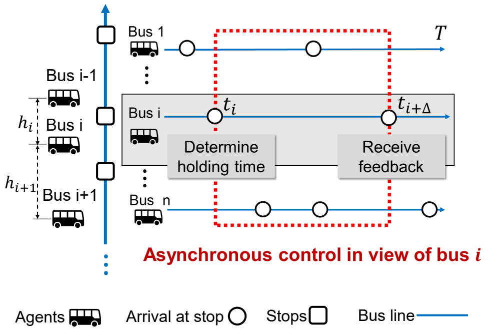

Figure 2: The vehicle control framework on a bus route. A circle indicates the event of a bus arriving at a bus stop. For bus b_i at time t_i, we denote by h_{i+1} the backward headway (time for the following bus b_{i+1} to reach the current location of b_i) and h_i the forward headway (time for b_i to reach the current location of b_{i-1}).

time as:

$$\Delta d_{i,t} = a_{i,t}\Delta T, \tag{1}$$

where ΔT is the maximum holding duration and $a_{i,t} \in [0, 1]$ is a strength parameter. We consider $a_{i,t}$ the action of bus b_i when arriving at a bus stop at time t. Here, ΔT is used to limit the maximum holding duration and avoid over-intervention. We model $a_{i,t}$ as a continuous variable in order to explore the near-optimal policy in an adequate action space. Note that no holding control is implemented when $a_{i,t} = 0$.

Reward: Despite that holding control can reduce the variance of bus headway and promote system stability, the slack/holding time will also impose additional penalties on both passengers (i.e., increasing travel time) and operators (i.e., increasing service operation hours). To balance system stability and operation efficiency, we design the reward function associated with bus b_i at time t as:

$$r_i^t = -(1-w) \times CV^2 - w \times a_{i,t}, \tag{2}$$

where $CV^2 = \frac{Var[h]}{E^2[h]}$ quantifies headway variability [Transportation Research Board, 2013] and $w \in [0, 1]$ is a weight parameter. Essentially, $E[h]$ is almost a constant given the schedule/timetable, and thus CV is mainly determined by $Var[h]$: a small CV indicates consistent headway values on the bus route, and a large CV suggests heavy bus bunching. The second term in Eq. (2) penalizes holding duration and prevents the learning algorithm from making excessive control decisions. This term is introduced because any holding decisions will introduce additional slack time and reduce operation efficiency. Overall, the goal of this reward function is to effectively achieve system stability with as few interventions as possible, with w serving as a balancing parameter.

The agent implements an action (i.e., determining holding time) when it arrives at a stop, and then it will receive feedback upon arriving at the next stop. In previous studies [Wang and Sun, 2020; Chen *et al.*, 2016], this dynamic process is considered a multi-agent extension of Markov decision processes (MDPs) [Littman, 1994] and referred to as Markov game $G = (N, \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where N denotes the number of agents, γ denotes the discount factor, $\mathcal{S}$ and $\mathcal{A} = \{A_1, \ldots, A_N\}$ denote the state space and the joint action space, respectively, and $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ represents the state transition function. However, this definition becomes inaccurate in the bus control problem, since agents rarely implement actions at the same time. To address this asynchronous issue, we introduce a modified formulation in which each agent maintains its own transition: $\hat{\mathcal{P}}_i : \mathcal{S}_i \times \hat{\mathcal{A}} \mapsto \mathcal{S}_i$, $\hat{\mathcal{A}} \subseteq \mathcal{A}$, where S_i is the observation of agent i. In this way, state transition observed by agent i does not necessarily depend on the actions from all the other agents at any particular time. The policy to choose an action is given by the conditional probability $\pi_{\theta_i} = p(A_i \mid S_i)$. Finally, the reward function for each agent can also be defined independently: $\mathcal{R} = \{R_1, \ldots, R_N\}$. While the independence assumption simplifies the problem, it raises a new challenge—how to effectively consider the actions from other agents, which is to be addressed in the following section.

4 Methods

As mentioned previously, traditional multi-agent models, such as Independent Q-Learning (IQL) [Tan, 1993], MADDPG [Lowe *et al.*, 2017] and QTran [Son *et al.*, 2019], cannot address the aforementioned asynchronous issue. To solve this problem, in this section we introduce a Credit Assignment framework for Asynchronous Control (CAAC) with shared parameters.

4.1 Credit Assignment Framework for Asynchronous Control (CAAC)

Following previous work, we adopt the basic actor-critic architecture, in which the critic is used to evaluate the effect of action from a single agent. To achieve reliable and efficient credit assignment in an asynchronous setting, we design an inductive critic following the scheme shown in Fig. 3.

The proposed inductive critic consists of an ego critic and an event critic: the ego critic is to evaluate the actor's policy, and the event critic is to account for the contribution from the actions of other agents. With this architecture, we expect CAAC to better quantify the undetermined impact from other agents and provide a more accurate credit assignment scheme for policy evaluation. We next introduce CAAC and the learning algorithm in detail.

Approximate Marginal Contribution with Event Critic

Inspired by the idea of inductive graph learning [Hamilton *et al.*, 2017], we propose to build a graph-based event critic

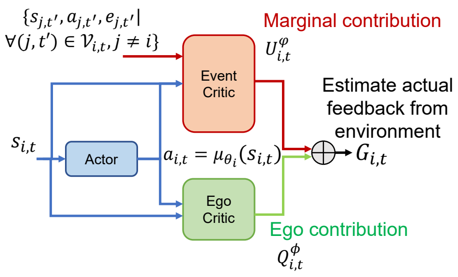
$\{s_{j,t'}, a_{j,t'}, e_{j,t'} |$
$\forall (j,t') \in \mathcal{V}_{i,t}, j \neq i\}$
Marginal contribution
$U^{\varphi}_{i,t}$
Event Critic
Estimate actual feedback from environment
$s_{i,t}$
Actor
$a_{i,t} = \mu_{\theta_i}(s_{i,t})$
$G_{i,t}$
Ego Critic
Ego contribution
$Q^{\phi}_{i,t}$

Figure 3: Architecture of CAAC.

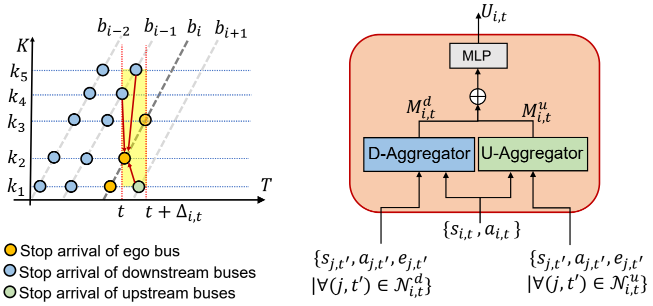
b_{i-2}
b_{i-1}
b_i
b_{i+1}
K
k_5
k_4
k_3
k_2
k_1
T
t
$t + \Delta_{i,t}$
● Stop arrival of ego bus
● Stop arrival of downstream buses
● Stop arrival of upstream buses
$U_{i,t}$
MLP
$\oplus$
$M^d_{i,t}$
$M^u_{i,t}$
D-Aggregator
U-Aggregator
$\{s_{i,t}, a_{i,t}\}$
$\{s_{j,t'}, a_{j,t'}, e_{j,t'} \mid \forall (j,t') \in \mathcal{N}^d_{i,t}\}$
$\{s_{j,t'}, a_{j,t'}, e_{j,t'} \mid \forall (j,t') \in \mathcal{N}^u_{i,t}\}$

Figure 4: Event Graph (EG) at time $t + \Delta_{i,t}$ (left) and the corresponding Event Critic (right) for agent/bus b_i. The dashed lines in EG show bus trajectories. We highlight the three incoming edges (in red) for event (i,t). D-aggregator and U-aggregator indicate aggregators for downstream events and upstream events, respectively.

to learn to approximate the marginal contribution from other agents. We first introduce the concept of event graph (EG) based on the spatiotemporal trajectory plot (see Fig. 4). We denote EG by $G(V,E)$, where V and E are the vertex set and edge set, respectively. We define a vertex (i,t) as the event when a bus b_i arrives at a bus stop at time t. Note that we ignore the index of bus stops when defining EG. We assume bus b_i arrives at the next stop at time $t + \Delta_{i,t}$, and introduce incoming edges for (i,t) by linking all arriving events (j,t') from other vehicles (i.e., $j \neq i$) within the time window $t < t' \leq t + \Delta_{i,t}$. The left panel of Fig. 4 illustrates how EG is built based on the trajectory diagram. Each node in the panel represents an arrival event of a bus. Taking the event (i,t) as an example, there are three incoming edges from other events. We denote the set of neighbors of event (i,t) by $\mathcal{N}_{i,t} = \{(j,t') \in V : j \neq i, t < t' \leq t + \Delta_{i,t}\}$.

Given an ego agent, the formulation of EG presents a neat and dynamic tool to account for actions from other agents, which allows us to conduct inductive graph learning and build a flexible policy evaluation block for asynchronous control. Specifically, we adopt graph attention neural network (GAT) [Veličković *et al.*, 2018] to inductively learn and approximate the undetermined impact from the actions of other agents. Considering the heterogeneous effects from upstream and downstream events, we introduce two separate GATs for upstream events and downstream events, respectively. In doing so, we divide $\mathcal{N}_{i,t}$ into two subsets: upstream event set $\mathcal{N}^u_{i,t} = \{(j,t') \in V : j > i, t < t' \leq t + \Delta_{i,t}\}$ and downstream event set $\mathcal{N}^d_{i,t} = \{(j,t') \in V : j < i, t < t' \leq t + \Delta_{i,t}\}$. The right panel of Fig. 4 summarizes the inductive block, which we call event critic (EC).

We next introduce the details of EC. We first define $e_{j,t'} = \left\{e^1_{j,t'}, e^2_{j,t'}\right\}, \forall (j,t') \in \mathcal{N}_{i,t}$ to capture the augmented information for the event critic, where $e^1_{j,t'}$ denotes the number of bus stops separating the two controls and it is normalized by the total number of stops in the system, and $e^2_{j,t'} = |j - i|$ is the number of vehicles in between b_i and b_j. We then aggregate information from upstream and downstream, respectively. Given the upstream events $\mathcal{N}^u_{i,t}$, the node features consist of two parts: (1) node features of other agents: $h_{j,t'} = (s_{j,t'}, a_{j,t'}, e_{j,t'})$, where $(s_{j,t'}, a_{j,t'})$ is the state-action pair of agent j at t'; (2) node feature of the ego agent $h_{i,t} = (s_{i,t}, a_{i,t}, \text{zeros\_like}(e_{j,t'}))$, where we add zeros to ensure equal feature length. Since EG is undirected, the graph attention is actually based on the relation between the ego event and its upstream events. We can then perform graph attention:

$$\alpha_{(i,t),(j,t')} = \frac{\exp\left(\sigma\left(f\left(W^a h_{i,t} \| W^a h_{j,t'}\right)\right)\right)}{\sum_{(j,t') \in \mathcal{N}^u_{i,t}} \exp\left(\sigma\left(f\left(W^a h_{i,t} \| W^a h_{j,t'}\right)\right)\right)} \tag{3}$$

$$h'_{i,t} = \sum\nolimits_{(j,t') \in \mathcal{N}^u_{i,t}} \alpha_{(i,t),(j,t')} W^a h_{j,t'}, \tag{4}$$

where σ is non-linear activation function and $\|$ represents concatenation, W^a is the trainable shared weight matrix, and f represents a single-layer neural network. Finally, we sum all the derived features as

$$M^u_{i,t} = \sum\nolimits_{k \in \left\{(i,t) \cup \mathcal{N}^u_{i,t}\right\}} \sigma\left(h_k\right). \tag{5}$$

The information aggregation M^d_i for downstream events $\mathcal{N}^d_{i,t}$ works the same way as the upstream. The summation of M^u_i and M^d_i is then fed to a feed-forward neural network to generate the final result of the event critic. Note that we omit self-attention and mask the aggregation output if there exist no upstream or downstream events for the ego agent (i.e., no contribution from other agents).

Deep Deterministic Policy Gradient with Inductive Critic

A critical issue in MARL with Deep Deterministic Policy Gradient (DDPG) is that the critic is likely to have an ambiguous evaluation on the contribution of each agent. The problem becomes even worse in an asynchronous setting where even the number of activated agents varies in each decision step. To address this issue, we implement multi-agent DDPG with a more rational credit assignment by combining the ego critic and event critic (see Fig. 3).

As the first component of the inductive critic, the ego critic is responsible for evaluating the ego policy. Based on the evaluation, we can derive policy gradient to maximize the following objective function as in DDPG:

$$\mathcal{J}(\theta) = \mathbb{E}\left[Q^{\phi}_{i,t}\left(s_{i,t}, \mu_{\theta}\left(s_{i,t}\right)\right)\right]. \tag{6}$$

The second part of the inductive critic is the event critic, which is used to approximate the marginal contribution from other agents. Finally, we approximate the actual system return for agent i using the sum of the output from ego critic and event critic:

$$G_{i,t} = Q^{\phi}_{i,t} + U^{\psi}_{i,t}. \tag{7}$$

Based on the standard DQN loss [Mnih *et al.*, 2015] following the Bellman equation, the proposed inductive critic can be trained by minimizing the following loss function:

$$\begin{aligned}\mathcal{L}(\phi,\psi) = \mathbb{E}\Bigg[&\left(r_{i,t} + \gamma G^{\text{tar}}_{i,t+\Delta_{i,t}} - G_{i,t}\right)^2 \\ &+ \beta\left(\mathbb{I}(N^u_{i,t} = \emptyset)\left\|M^u_{i,t}\right\|_2 + \mathbb{I}(N^d_{i,t} = \emptyset)\left\|M^d_{i,t}\right\|_2\right)\Bigg],\end{aligned} \tag{8}$$

where $r_{i,t}$ is the reward obtained by agent i given its action $a_{i,t}$, $G^{\text{tar}}_{i,t+\Delta_{i,t}}$ denotes the cumulative return from the next decision step $t + \Delta_{i,t}$ in the view of agent i, which is estimated through the target network [Mnih *et al.*, 2015], and $\mathbb{I}(\cdot)$ is an indicator function. Note that in the case where an event has no upstream/downstream events, we additionally impose L2 norm with hyperparameter $\beta = 0.1$. Thus, $\left\|M^u_{i,t}\right\|_2$ and $\left\|M^d_{i,t}\right\|_2$ regularize W^a in the event critic block, which can facilitate inductive learning. In training, we sample from the interaction between agents and the system to approximate the expectation of above objective functions. In the execution phase, only local observations are used for actor network; thus, the overall CAAC is very flexible for real-world applications.

5 Evaluation

We evaluate the performance of the proposed method based on real-world data. Four bus routes (R1-R4) in an anonymous city are selected, with true passenger demand derived from smart card data. The four routes are all trunk services covering more than 15 km with over 40 bus stops along the route. Table 1 lists the basic statistics of the routes, including the number of services per day, the number of stops, route length, the mean and standard deviation (std) of headway at the departure terminal.

	services	stops	length (km)	mean (sec)	std (sec)
R1	59	46	17.4	874	302
R2	72	58	23.7	745	307
R3	57	61	23.2	931	354
R4	55	46	22.5	955	351

Table 1: Basic information of bus lines.

5.1 Simulator for Bus Route Operation

We first develop and calibrate the bus simulator to reproduce the patterns of real-world operation. In this simulation, the alighting and boarding times per passenger are set to $t_a = 1.8$ s/pax and $t_b = 3.0$ s/pax, respectively. To simulate the uncertainty of road conditions, buses are given a random speed $v \times \mathcal{U}(0.6, 1.2)$ km/h when travelling between every two consecutive stops, where v is set to 30 km/h and $\mathcal{U}$ denotes a continuous uniform distribution. The capacity of the bus is set to 120 pax. Fig. 5 (a) and (b) show the simulated boarding time and actual boarding time from smart card (tap-in for boarding and tap-out for alighting) data and the simulated journey time and actual journey time, respectively, for service R1. The Pearson correlations for these two plots are 0.999 and 0.983, respectively, suggesting that our simulator is consistent with the real-world operation.

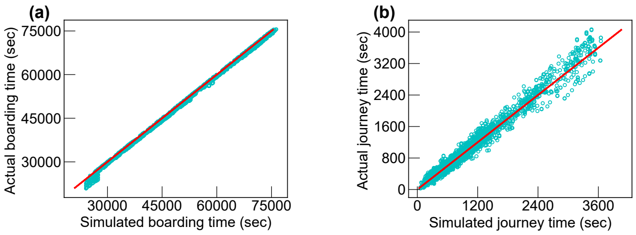

Figure 5: Simulated values v.s. true values from smart card data on R1: (a) boarding time (i.e., smart card tapping-in time), and (b) journey time (i.e., duration between tapping-in and tapping-out).

5.2 Experimental Settings

We set hyper-parameter $w = 0.2$ in reward function Eq. (2) to place priority on system stability. We start our experiment by training the model on R1 for 250 episodes. In the execution phase, we test the performance of the tuned RL models on R1 and then evaluate the model transferability by applying R1's models on R2/R3/R4 directly without retraining. The proposed **CAAC** framework is compared with the following baseline models, including both traditional headway-based control models and state-of-the-art MARL methods: i) **No control (NC)**: NC is considered a naive baseline where no holding control is implemented. ii) **Forward headway-based holding control (FH)** [Daganzo, 2009]: the holding time is $d = \max\{0, \bar{d} + g(H_0 - h^-)\}$, where H_0 is the desired departure headway, h^- is the forward headway of the arriving bus, $\bar{d}$ is the average delay at equilibrium, and $g > 0$ is a control parameter. We use the same parameters as in Daganzo [2009]. iii) **Independent Actor-Critic (IAC)** [Lillicrap *et al.*, 2016]: we implement DDPG in the IAC setting to examine the performance where the agent completely overlooks the impact from other agents. IAC can be considered a special case of CAAC with no event critic. iv) **MADDPG** [Lowe *et al.*, 2017]: we implement MADDPG as a state-of-the-art baseline without special consideration for the asynchronous setting. In MADDPG, the ego bus considers all other buses when performing actions. In the implementation, we fill zeros for the actions from inactivated agents. Besides, to achieve transferability, both the critic network and the actor network are shared among all agents.

All models are implemented with python and PyTorch 1.7.0 on Ubuntu 18.04 LTS, and experiments are conducted on a server with 256GB RAM. We use the following indicators to evaluate model performance: i) **Average holding time (AHT)**, which characterizes the degree of intervention; ii) **Average waiting time (AWT)**, which evaluates the severity of bus bunching; iii) **Average journey time (AJT)**, which quantifies the average journey duration from boarding to alighting for all trips; iv) **Average travel time (ATT)**, which quantifies the average travel time for each bus from the departure terminal; and v) **Average occupancy dispersion (AOD)**, which evaluates how balanced the occupancy is. Note the dispersion is computed as a variance-to-mean ratio of the number of onboard passengers. We expect to see a large AOD value when bus bunching happens, since the number of onboard passengers will be highly imbalanced in those situations.

5.3 Results

Execution Results on Trained Route (R1)

We first evaluate model performance on R1 on which all the MARL models are trained. Fig. 6 presents detailed stop-wise

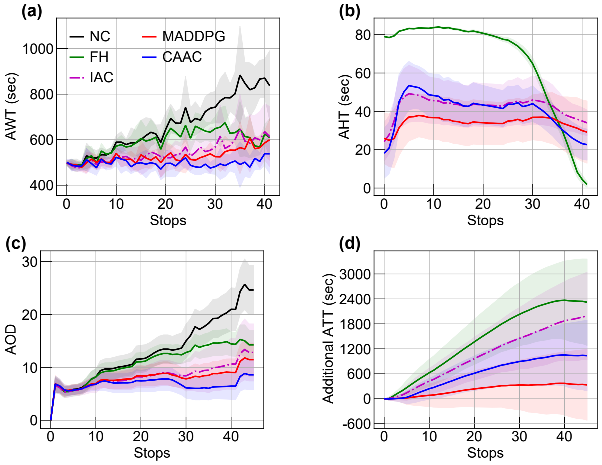

Figure 6: Stop-wise performance comparison: (a) average waiting time; (b) average holding time; (c) average occupancy dispersion; (d) additional travel time from departure terminal w.r.t. NC.

performance comparison. Overall, the result suggests that RL-based models perform better in reducing bus bunching at a lower cost than traditional headway-based control FH. We believe it is because of the long-term reward consideration and informative guidance from state definition in RL models, which are particularly beneficial to efficient transit control. In particular, the proposed CAAC performs the best in stabilizing the system: it shows the smallest AWT and AOD values, while imposing more control interventions than MADDPG. This is mainly due to the fact that MADDPG does not distinguish its own contribution from that of others. Consequently, the reward in MADDPG is mainly determined by the agent's own control actions, leading to biased strategies. IAC can be considered a special case of CAAC, with the event critic component removed. Our results show that IAC is also less effective than CAAC, since the impact from other agents is completely ignored in IAC and therefore it suffers from the uncertainty of the system dynamics.

Fig. 7 shows the trajectory plot colored by occupancy (i.e., number of onboard/capacity) under different control policies based on one random seed. As can be seen, the real-world operation provides a challenging scenario where it is hard to maintain headway regularity over the whole day without any intervention (i.e., NC). Notably, all MARL strategies seem to bring improvements at the upstream segment (i.e., 0-5 km). However, we see that CAAC clearly outperforms IAC and MADDPG in maintaining headway consistency over the downstream segment.

Execution on Untrained Routes (R2/R3/R4)

When applying RL in real-world applications, one critical challenge is to ensure model transferability. As we adopt inductive graph learning in designing the event critic, we expect that trained model on one service can be also applied/generalized to other unseen services. To examine the transferability, we apply the models trained on R1 directly on R2/R3/R4 without retraining. Note that the configurations of R2/R3/R4 in terms of bus stop location, number of stops,

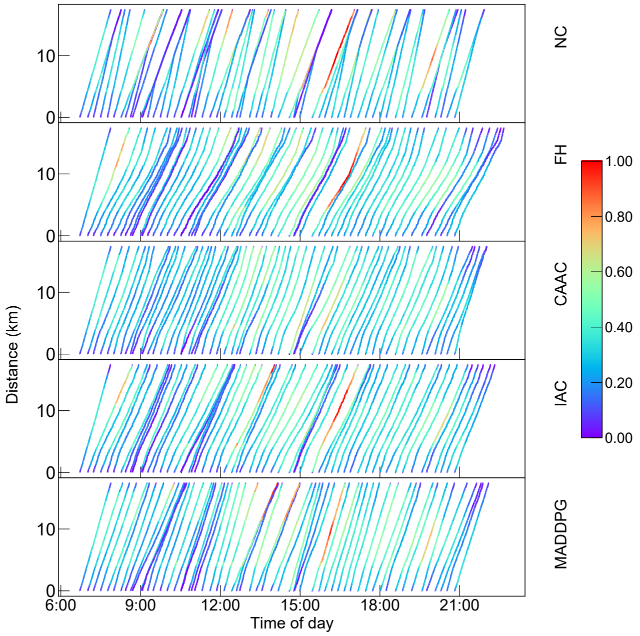

Figure 7: Bus trajectories under different holding strategies.

and number of buses (i.e., agents) are different from that of R1. In addition, the default timetable and the demand pattern on R2/R3/R4 also differ substantially from those of R1. Thus, this experiment presents an unseen/unfamiliar environment for those RL models trained on R1. To better compare the models, we randomly scale demand with multiple random seeds and runs the model 10 times for each random seed.

Table 2 summarizes the overall performance of different models. For IAC, agents are trained in the environment ignoring the impact from other agents on the system. When transferred to another scenario, the performance greatly deteriorates due to the variation in dynamics from other agents. For MADDPG, it only partially considers actions from other agents given the uncertainty in agent states (i.e., activated or not). As a result, MADDPG performs worse on R2/R3/R4 than on R1. In contrast, the proposed CAAC shows superior transferability when applied to R2/R3/R4 in terms of system stability, maintaining comparable performance to that of R1. The above transferability analysis reinforces the point that, in an asynchronous setting, the training process clearly benefits from taking those undetermined events into consideration. As an example, CAAC demonstrates remarkable inductive power and transferability to unseen scenarios.

6 Conclusion

In this paper, we develop an asynchronous multi-agent reinforcement learning model to improve dynamic bus control and reduce bus bunching. In particular, we propose CAAC—a credit assignment framework to address the asynchronous issue in multi-agent control problems, which is often overlooked in previous studies. To effectively consider the impact/contribution of other agents, we design an inductive critic by combining an ego critic and an event critic. This design can be easily embedded in the actor-critic framework,

Method	Performance on R2/R3/R4			
	AHT (sec)	Δ AWT (sec)	Δ AJT (sec)	Δ AOD
NC	-/ - / -	625 / 757 / 723	1415 / 1389 / 1109	18.4 / 13.1 / 9.9
FH	67/ 67 / 68	-87/ -70 / -36	+614 / +780 / +480	-5.1 / -2.4 / -2.5
CAAC	42/ 40* / 41*	**-166/ -161 / -127**	+251* / +336* / +192*	**-9.9 / -5.9 / -4.9**
IAC	40*/ 42 / 42	-128/ -128 / -94	+254 / +360 / +202	-7.5 / -4.7 / -4.0
MADDPG	**37/ 35 / 36**	-132*/ -145* / -111*	**+202 / +223 / +130**	-7.7* / -5.0* / -3.7*

Table 2: Execution performance when applying models trained on R1 directly on R2/R3/R4 without retraining. Model performance is evaluated using: 1) average holding time (AHT), and changes in—2) average waiting time (ΔAWT), 3) average travel time (ΔATT), and 4) average occupancy dispersion (ΔAOD)—w.r.t. the NC baseline. The best and second best results are highlighted in bold and with asterisk ($*$), respectively.

and it turns out to be efficient and effective in policy training. Overall, our results demonstrate that the proposed CAAC framework is simple but very effective in characterizing the marginal contribution from other agents in bus holding as an ASMR problem. The graph neural network-based inductive critic offers additional generalization and inductive power, which allows us to apply a well-trained model directly on a new bus route.

There are several directions for future research. While the current model trains the agents with randomly scaled demand, we suspect that the performance of agents can be further improved using more informative/reasonable demand patterns. Another future research direction is to incorporate more traffic entities (e.g., traffic signals) with the inductive architecture and develop a more comprehensive traffic control policy such as signal priority for buses. Finally, the method can be generalized to other applications with similar spatiotemporal asynchronous patterns.

Acknowledgments

This research is supported by the Fonds de Recherche du Québec - Société et Culture (FRQSC) under the NSFC-FRQSC Research Program on Smart Cities and Big Data, and the Canada Foundation for Innovation (CFI) John R. Evans Leaders Fund.

References

[Alesiani and Gkiotsalitis, 2018] Francesco Alesiani and Konstantinos Gkiotsalitis. Reinforcement learning-based bus holding for high-frequency services. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 3162–3168, 2018.

[Chang *et al.*, 2004] Yu-Han Chang, Tracey Ho, and Leslie P Kaelbling. All learning is local: Multi-agent learning in global reward games. In *NeurIPS*, pages 807–814, 2004.

[Chen *et al.*, 2016] Weiya Chen, Kunlin Zhou, and Chunxiao Chen. Real-time bus holding control on a transit corridor based on multi-agent reinforcement learning. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 100–106, 2016.

[Chen *et al.*, 2020] Chacha Chen, Hua Wei, Nan Xu, Guanjie Zheng, Ming Yang, Yuanhao Xiong, Kai Xu, and Zhenhui Li. Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control. In *AAAI*, pages 3414–3421, 2020.

[Daganzo and Ouyang, 2019] Carlos F Daganzo and Yanfeng Ouyang. *Public Transportation Systems: Principles of System Design, Operations Planning and Real-time Control*. World Scientific, 2019.

[Daganzo, 2009] Carlos F Daganzo. A headway-based approach to eliminate bus bunching: Systematic analysis and comparisons. *Transportation Research Part B: Methodological*, 43(10):913–921, 2009.

[Foerster *et al.*, 2018] Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *AAAI*, pages 2974–2982, 2018.

[Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, pages 1024–1034, 2017.

[Lee *et al.*, 2020] Jongmin Lee, Byung-Jun Lee, and Kee-Eung Kim. Reinforcement learning for control with multiple frequencies. In *NeurIPS*, 2020.

[Lillicrap *et al.*, 2016] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *ICLR*, 2016.

[Littman, 1994] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *ICML*, pages 157–163. 1994.

[Lowe *et al.*, 2017] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *NeurIPS*, pages 6379–6390, 2017.

[Menda *et al.*, 2018] Kunal Menda, Yi-Chun Chen, Justin Grana, James W Bono, Brendan D Tracey, Mykel J Kochenderfer, and David Wolpert. Deep reinforcement learning for event-driven multi-agent decision processes. *IEEE Transactions on Intelligent Transportation Systems*, 20(4):1259–1268, 2018.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[Rashid *et al.*, 2018] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML*, page 4295–4304, 2018.

[Seman *et al.*, 2019] Laio Oriel Seman, Luiz Alberto Koehler, Eduardo Camponogara, Lucas Zimmermann, and Werner Kraus. Headway control in bus transit corridors served by multiple lines. *IEEE Transactions on Intelligent Transportation Systems*, 21(11):4680–4692, 2019.

[Son *et al.*, 2019] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *ICML*, page 5887–5896, 2019.

[Sunehag *et al.*, 2018] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinícius Flores Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *AAMAS*, pages 2085–2087, 2018.

[Tan, 1993] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *ICML*, pages 330–337, 1993.

[Transportation Research Board, 2013] Transportation Research Board. *Transit Capacity and Quality of Service Manual, Third Edition*. The National Academies Press, Washington, DC, 2013.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[Wang and Sun, 2020] Jiawei Wang and Lijun Sun. Dynamic holding control to avoid bus bunching: A multi-agent deep reinforcement learning framework. *Transportation Research Part C: Emerging Technologies*, 116:102661, 2020.

[Wang *et al.*, 2020] Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. Shapley q-value: A local reward approach to solve global reward games. In *AAAI*, pages 7285–7292, 2020.