A GENERAL SCENARIO-AGNOSTIC REINFORCEMENT LEARNING FOR TRAFFIC SIGNAL CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

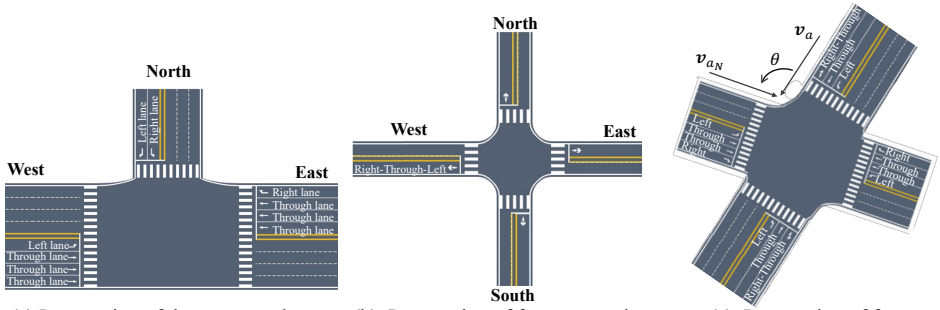
Reinforcement learning has been recently adopted to revolutionize and optimize traditional traffic signal control systems. Existing methods are either based on a single scenario or multiple independent scenarios, where each scenario has a separate simulation environment with predefined road network topology and traffic signal settings. These models implement training and testing in the same scenario, thus being strictly tied up with the specific setting and sacrificing model generalization heavily. While a few recent models could be trained by multiple scenarios, they require a huge amount of manual labor to label the intersection structure, hindering the model's generalization. In this work, we aim at a *general* framework that could eliminate heavy labeling and model a variety of scenarios *simultaneously*. To this end, we propose a GEneral Scenario-Agnostic (GESA) reinforcement learning framework for traffic signal control with: (1) A general plug-in module to map all different intersections into a unified structure, freeing us from the heavy manual labor to specify the structure of intersections; (2) A unified state and action space to keep the model input and output consistently structured; (3) A large-scale co-training with multiple scenarios, leading to a generic traffic signal control algorithm. In experiments, we demonstrate our algorithm as the first one that can be co-trained with seven different scenarios without manual annotation, and get **13.27%** higher rewards than benchmarks. When dealing with a new scenario, our model can still achieve **9.39%** higher rewards. The code and scenarios are available here under an anonymous Github page.

1 INTRODUCTION

Although urbanization has been evolving for decades, traffic congestion remains severe in major cities, which lays up potential risks, such as traffic accidents or unnecessary time and resource wastage. An efficient traffic signal control (TSC) system is undoubtedly a cost-effective mean of alleviating congestion. Thus, how to optimize traffic signals has always been the spotlight for urban efficiency and sustainability (Wei et al., 2019c).

Traditional TSC systems, such as SCOOT (Hunt et al., 1982) and SCATS (Lowrie, 1990), are widely deployed in hundreds of cities worldwide, whose traffic signal plans are manually designed according to rules from expert knowledge and transportation engineering. For example, Webster's Equation (Koonce & Rodegerdts, 2008), one of the most widely-used methods, directly uses the traffic volumes to calculate the optimal cycle length and phase split for a single intersection to minimize the total travel time at this intersection. However, those methods may be powerless when handling dynamic traffic networks without well-observed traffic flow patterns. To this end, reinforcement learning (RL) (Steingrover et al., 2005; Van der Pol & Oliehoek, 2016; Wei et al., 2018; 2019a) has been preferably adopted into the TSC domain since it is a learning-based method with higher automation. Such a trial-and-error paradigm based on the traffic simulator has demonstrated better performance than transport engineering-based methods (Yau et al., 2017).

The recent RL-based TSC models can be roughly divided into two categories based on the scenarios where the training and testing are conducted. A scenario is usually a simulation environment that contains a set of intersections: (1) Single-scenario RL, whose training and testing need to be on the same scenario separately (El-Tantawy et al., 2013; Van der Pol & Oliehoek, 2016; Zheng et al., 2019). However, the model will be unusable or perform badly in a new scenario. For example in



(a) Intersection of three approaches, with 2-4 entering lanes, and 2 movements on each approach

(b) Intersection of four approaches, with 1 entering lane of right-through-left movement on each approach

(c) Intersection of four approaches, with 4 entering lanes, and 3 movements, yet with an arbitrary angel

Figure 1: Three intersections with different structures in terms of approach, movement, and lane

Fig. 1, these methods might be trained and tested in the same scenario with 4×4 three-approach intersections of Fig. 1(a), but these methods will ill-perform in another new scenario with mixed intersections of Fig. 1(a) and 1(b). (2) Multi-scenario RL, where training is conducted in multiple scenarios and testing could be in different scenarios. For example, Meta RL (Zang et al., 2020) is proposed to train a TSC system with multi-scenarios. The model can easily transfer to other scenarios. However, the existing multi-scenario RL models need heavy manual labor to annotate the structure of intersections during the training.

However, the above methods are trained with several pre-defined and fixed scenarios whereas cannot gain the generalization capability without labeling, which limits the application of RL-based methods in the real world. Current RL-based TSC methods can exploit the various traffic flows generated by the simulator to make the model effective in training scenarios, but finding a low-cost universal method with promising transferability meanwhile is still a research gap.

As a result, the existing methods still face tremendous challenge to jump out from the simulation and implement in the real cities. This is known as *sim2real* challenge. The challenge mainly come from the wide gap between the real complex cities and the simplified simulation systems. In the real world, the intersection structure could be rather versatile in terms of different setting of approaches (*i.e.*, north, south, east, west), movements (*i.e.*, left, right, through), and lanes (*e.g.*, two through lanes, one right-through lane). As shown in Fig. 1(a) to 1(c), an intersection could have a different amount of approaches; Within an approach, there could be different combinations of movements; A lane could also combine different movements. However, most of the existing methods only consider a standard intersection with four approaches and three lanes (right, through, and left) within each approach, shown in Fig. 2(a). This largely limits the model generalization.

To conclude, a qualified TSC approach needs high generalization and effectiveness: it should handle various intersections and be able to transfer to other unseen targets easily and with low cost.

To narrow the *sim2real* gap significantly and get more ready to be implemented in real cities, in this paper, we provide our novel solution, i.e., a **GE**neral **S**cenario-**A**gnostic (GESA) reinforcement learning framework, for the TSC task, which is, to our best knowledge, the first work that pursuits high generability and co-trains multiple scenarios without labels. Specifically, to co-train in multiple scenarios with various intersections, the vectors with approach spatial information are employed to map shape-odd and complex intersections into the standard intersection. Then, the mapped intersections are used to generate the characteristic information of each traffic movement and the phase of the traffic lights in a specific order. Finally, we extend the original FRAP (short for Flipping and Rotation and considering All Phase configurations) (Zheng et al., 2019) to a policy gradient-based framework, which can facilitate the model coverage and is compatible with different intersections.

The contributions are summarized in three-fold: (1) we present a general plug-in module to map the intersections into a unified structure, freeing us from the heavy manual labor work to specify

the intersection structure and making it possible for large-scale co-training under multiple different scenarios. (2) Accordingly, we design a unified state and action space to keep the model input and output structure consistent for more general capabilities. Moreover, the GESA can adapt to various unseen scenarios and achieve promising performance without re-training. (3) We build two realistic scenarios, together with five public scenarios, where we co-train and validate the GESA with prudent experiments. All these lead us closer to the ultimate goal: to implement RL-based TSC in the reality.

The rest of the paper is organized as follows. In Section 2, we review related work for TSC. Section 3 provides the preliminaries and Section 4 introduces the model thoroughly. Experiments are given in Section 5. The conclusion are presented in Section 6.

2 Related Work

We will review the existing methods for TSC from the two methodology stakes: transportation engineering-based methods and reinforcement learning-based methods.

Traditional traffic signal control. Until now, traditional methods are still widely used in the real world. Most of them depend on strong assumptions and manually-specified rules. The fixed-time (Roess et al., 2004) method decided the traffic signal plan as a fixed cycle length, which is not flexible enough and cannot automatically adapt its logic to the actual situation. Actuated control model (Fellendorf, 1994; Mirchandani & Head, 2001) instead used pre-defined sets of rules to decide whether to adjust the current phase. SCOOT (Hunt et al., 1982) and SCATS (Lowrie, 1990) pre-defined several signal plans and selected the best of them according to the internal measurements. Recently, MaxPressure (Varaiya, 2013; Kouvelas et al., 2014) was designed to dynamically optimize vehicle travel time and achieves expressive performance. It balances the queue length between incoming lanes and outgoing lanes to reduce the risk of over-saturation. In this way, its logic can be dynamically changed according to the actual situation. All these methods are simple and interpretable, which can be applied to the vast majority of traffic signal systems in the real world, but their assumption dependencies are relatively simple and rigid. Actual situations may be not satisfied with them, so the traditional methods usually cannot guarantee optimal results.

Reinforcement learning-based traffic signal control. Recently, reinforcement learning methods have been deeply researched for the TSC task. In (Prashanth & Bhatnagar, 2011; Casas, 2017), all intersections in a scenario are controlled by a global agent, where the state and action space are joint. The methods in references (Xu et al., 2013; El-Tantawy et al., 2013; Van der Pol & Oliehoek, 2016) handle the states separately, but they are still modeled as a joint action. These methods cannot be trained only once and then directly transferred to a new scenario. Thus, if one needs to adapt the well-trained model to an unseen intersection, model re-training in this new scenario is necessary. Such approaches have extended time and money costs, and their performance cannot be guaranteed. In recent years, independent RL becomes popular because of its low computational cost, feasibility, and scaled up easily. It is more suitable for application in the real world. In (Gao et al., 2017; Wei et al., 2018), the states are treated as an image for each intersection and the actions are handled independently. (Nishi et al., 2018; Wei et al., 2019b; Zhang et al., 2020) introduce communication to coordinate the actions between intersections. (Zheng et al., 2019) present a flexible network FRAP that can be used with different intersections based on phase competition. Based on FRAP, Chen *et al.* (Chen et al., 2020) add the pressure observation into the states and rewards to extend FRAP to thousand lights control scenarios. Then, Zhu *et al.* (Zhu et al., 2022) introduce a multi-task pipeline, i.e., MTLight, to predict next step states, which can express more general underlying characteristics. But all these methods need to be trained and evaluated under the same scenarios to get better performance, which limits the development of the algorithm from simulation to reality. The most related work is MetaLight (Zang et al., 2020), which combines Meta RL (Finn et al., 2017; Yu et al., 2020) into FRAP framework, such that it could also be trained under multi-scenarios simultaneously and thus can be transferred to another scenario without re-training. However, it still needs heavy manual work to label the intersection structure. Moreover, during experiment, we found the multi-scenatios that MetaLight used for training are still quite homogeneous, and the Meta Learner is only trained with two scenario each time, which heavily limits the model's generalization.

In this paper, to overcome these limitations, we design a flexible plug-in module to automatically process the various structures of intersections and extend the FRAP model to work in a unified state and action space, making it possible to co-train with multi-scenarios without labeling needed.

3 PRELIMINARIES

In this paper, the TSC is conducted at a single intersection. As mentioned before, several intersections constitute a scenario. In the following, the problem of this work is formulated formally, and then, for a particular intersection, the adopted terms are declared.

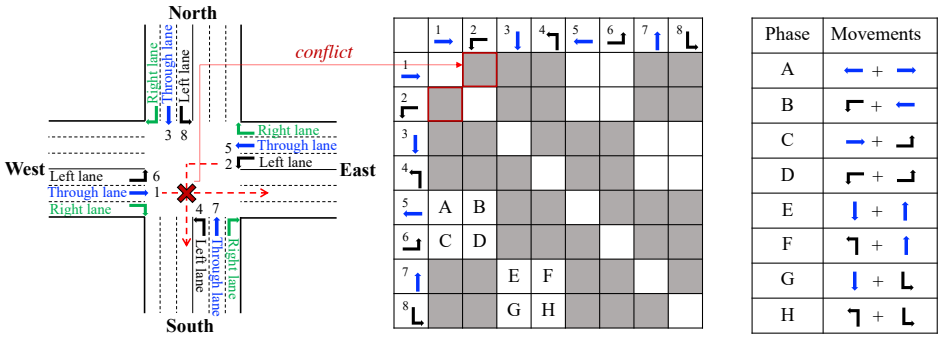
Problem Statement: The TSC is a sequential decision task characterized by the Markov decision process (MDP), which can be formulated as a single-agent RL problem $< \mathcal{S}, \mathcal{A}, \mathcal{R}, \gamma, \pi >$. Given the state observation set $\mathcal{S}$, which includes all related information such as traffic volume and current phase, action set $\mathcal{A}$, the reward function $\mathcal{R}$, and discount factor γ, the goal for each intersection agent i at each timestep t is to find a policy π that maximizes the expected return $G^t := \sum_{l=0}^{\infty} \gamma^l r^{t+l}$.

Definition 1. ***Intersection***: *The types of intersections are versatile in the real world. A standard intersection is shown in Fig. 2.(a), and the others can be treated as a variant of it. Each intersection has several approaches, where approaches can be categorized as the entering approach and the exiting approach. There are four entering approaches in Fig. 2(a) with lanes labeled, located in east, west, north, and south directions, respectively. There are also four exiting approaches in Fig. 2(a), which are left blank. In the real world, most intersections consist of three or four approaches.*

Definition 2. ***Entering approach***: *In each entering approach, there should be at least one entering lane. In Fig. 2(a), each approach has three entering lanes: right, through, and left.*

Definition 3. ***Traffic movement***: *A traffic movement is defined as a vehicle moving from an entering approach to an exiting approach. Fig. 2(a) shows eight movements (left and through) to be signal-controled and four right movements that are free (since right lanes do not conflict with other lanes). The conflict matrix of the eight movements is shown in Fig. 2(b). A movement can have more than one entering lane and an entering approach can also contain several movements.*

Definition 4. ***Phase***: *A traffic phase is a combination of movement signals. As illustrated in Fig. 2(c), there are eight phases in the standard intersection, and each phase combines two traffic movements without conflicts. Noted that not all the phases are available for the actual intersections.*



(a) A standard intersection with four approaches and three lanes (right, through, left) within each approach

(b) Conflict matrix for eight movements (symmetric), with unavailable phase in grey

(c) Eight phases from non-conflicting movements

Figure 2: A standard intersection structure, conflict matrix, and available traffic signal phases

4 METHOD

In this section, the detail of the proposed GESA model will be given. Section 4.1 and Section 4.2 will introduce (1) the well-elaborated plug-in module that unifies all intersections and (2) the unified state and action design, respectively, in a general way. Section 4.3 will give setting details for RL.

4.1 TOWARD TO A UNIFIED STRUCTURE

To unify different intersections, the plug-in module is designed, which maps the intersections with arbitrary structures into the standard one, makeing each entering lane only have one main movement.

In contrast to the standard intersection in Fig. 2(a), the structures of the real-world intersections are arbitrary even shape-odd. Fig. 1(a) shows a three-approach intersection. Compared with the standard intersection, its south approach and the through lane in the north approach are absent. In Fig. 1(b), each entering lane has multiple traffic movements, which can be frequently observed in urban streets. The approach directions are irregular in Fig. 1(c), which is also common.

4.1.1 TO UNIFY THE APPROACHES

Actually, the unification of approaches is to align their directions in the target intersection and the standard intersection, respectively. We provide the approach-unifying process, shown in Algorithm 1. For particular intersection i, we first retrieve the position vectors (e.g., geo-locations) for the approaches and count their numbers. Then, the position vectors are employed to calculate the angles between the approaches: The angle $\theta_{a,a'}$ between approaches a and a' is calculated by $\theta_{a,a'} = \arccos(\frac{\mathbf{v}_a \cdot \mathbf{v}_{a'}}{\|\mathbf{v}_a\| \cdot \|\mathbf{v}_{a'}\|})$, where $\mathbf{v}_a$ and $\mathbf{v}_{a'}$ represent the spatial locations of approaches a and a', respectively. Finally, the intersection is matched to the standard one according to the angles, as shown in Fig. 1(c): (1) An approach is selected to associate with the north approach a_N in the standard intersection. (2) The other approaches $\{a\}$s, denoted as $\mathbf{a}$, are matched approximately to the standard intersection's approaches according to the related angles. Specifically, if the angle between the a_N and a is between 45° to 135°, a will be matched to the east approach a_E. South and west approaches are matched similarly. If each direction matches no more than one approach, the matching process will be successful. It will be treated as fail if all the approaches are traversed iteratively without success, since such an intersection is usually odd.

Algorithm 1 To unify the approach directions

Input: The number of approaches of target intersection $|\mathbf{a}|$; The position vectors of the approaches $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{|\mathbf{a}|}\}$;
Initialization: Indicator of success in unification *Flag=False*;
for a^* in $\{a\}$s **do**
  Initialization: Approach-direction set $\mathcal{M} = \{m_N : \text{set}(a^*), m_S : \text{set}(), m_W : \text{set}(), m_E : \text{set}()\}$;
  for a in $\{a\}$s and $a \neq a^*$ **do**
    $\theta_{a,a^*} = \arccos(\frac{\mathbf{v}_a \cdot \mathbf{v}_{a^*}}{\|\mathbf{v}_a\| \cdot \|\mathbf{v}_{a^*}\|})$
    if $\mathbf{v}_a \times \mathbf{v}_{a^*} < 0$ **then**
      $\theta_{a,a^*} = 360° - \theta_{a,a^*}$
    end if
    if $45° \leq \theta_{a,a^*} < 135°$ **then**
      $m_E.\text{add}(a)$
    else if $135° \leq \theta_{a,a^*} < 225°$ **then**
      $m_S.\text{add}(a)$
    else if $225° \leq \theta_{a,a^*} < 315°$ **then**
      $m_W.\text{add}(a)$
    else
      $m_N.\text{add}(a)$
    end if
  end for
  if length(m_N)<2 and length(m_S)<2 and length(m_W)<2 and length(m_E)<2 **then**
    Flag=True;
    Break;
  end if
end for
Output: *Flag*, $\mathcal{M}$.

4.1.2 TO UNIFY THE MOVEMENTS

The objective of movement unification is to achieve one-to-one matching between lanes and movements and handle unavailable phases. For the case that one lane is associated with multiple movements (e.g., Fig. 1(b)), we can delineate their priority according to the traffic rules: (1) Through movement is with the highest priority. (2) Turning left has priority over turning right. (3) Turning right commonly is free and needs no signal control. Thus, we can take all entering lanes as through lanes in Fig. 1 (b). Finally, eight traffic signal phases can be generated as shown in Fig. 2 (c). Red signals are adopted to fill the unavailable phases.

All the above rules are integrated into a plug-in module, which can be placed in the simulation scenarios' initialization stage while eschewing extensive manual labor to label the distinct intersections.

4.2 TOWARD TO A GENERAL MODEL

To ensure that the RL model gains better transferability, it should be compatible with different kinds of intersections. Essentially, the in-out dimensions of the model need to be constant regardless of the intersections. Thus, it is crucial to unify the agent's state space and action space.

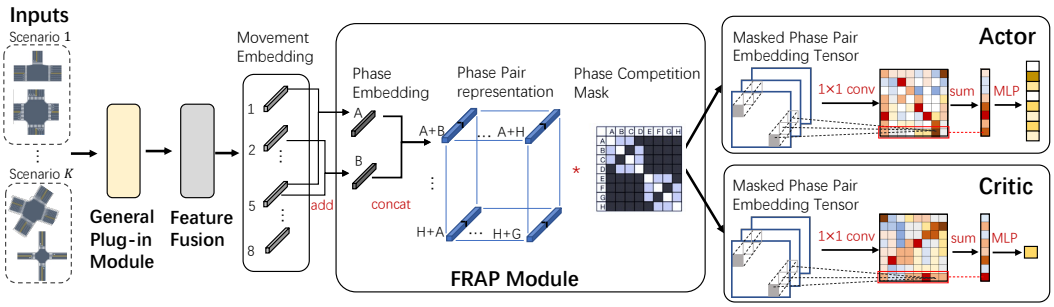


Figure 3: Network design in our framework

4.2.1 A GENERAL STATE SPACE

For the agent state space, with no need to control the right-turn behaviors, we first gather the observations with eight groups according to the movements they belong in the standard intersection (as Fig. 2(a)). There are multiple demands for each group, such as queue length, flow, and occupancy. Second, the movements of multiple entry lanes may be the same. In this case, we aggregate the average observations in the lanes to represent the demand for the movement. Third, considering that some traffic movements may be absent compared with the standard intersection, we use zero padding in the missing group demands and add a binary indicator in the corresponding states.

4.2.2 A GENERAL ACTION SPACE

The agent action is to choose the phase for the next time interval. For the action space, we reserve all eight phases and the order of these phases is fixed (same as Fig. 2(c)), letting the agent have the same logic to select phases. However, not all eight phases are always available, e.g., phases E and F in Fig. 2.(c) do not appear in the intersection in Fig. 1.(a). To solve this, we use mapped intersection structures to identify unavailable phases, and add a mask to let the agent ignore them.

4.2.3 LEARNING

Our unified design can enable most RL models to handle arbitrary intersections. For instance, if adopting our unified design, a generic Proximal Policy Optimization (PPO) (Schulman et al., 2017) can be co-trained with multiple scenarios by Asynchronous Advantage Actor-Critic (A3C) (Mnih et al., 2016).

We modify the FRAP to a policy gradient-based design. Our framework is shown in Fig. 3. We denote in the i-th intersection from the k-th scenario, the j-th movement's n-th feature as $f^m_{i,j,k,n}$, where $i \in \mathbb{R}^{I_k}$, $k \in [1, \ldots, K]$, and $j \in [1, \ldots, 8]$ as eight movements.

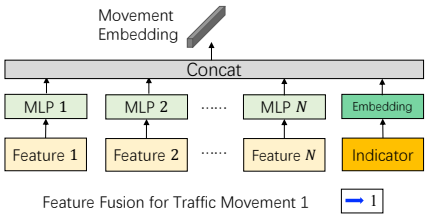


Figure 4: A traffic movement embedding

Firstly, via the General Plug-In (GPI) module, arbitrary intersections from various scenarios are mapped into the standard one. Thus, the feature will be scenario-agnostic and intersection-agnostic.

$$\tilde{f}^m_{i,j,n} = GPI(f^m_{i,j,k,n}) \tag{1}$$

Then, the embedding for the j-th traffic movement's all features is obtained by Multi-Layer Perceptron (MLP) and concatenation (denoted as operator $\|$), as shown in Fig. 4. Besides, the last feature "indicator" (binary) denotes whether this movement exists in this intersection.

$$\bar{\mathbf{f}}^m_{i,j} = \|_{n=1}^{N+1} MLP(\tilde{f}^m_{i,j,n}) \tag{2}$$

Secondly, the calculated movement embedding $\bar{\mathbf{f}}^m_{i,j}$ is fed into the FRAP module. (1) Add: FRAP adds the embedding of movements as a phase representation. For the l-th phase which consists of

movements $j, j', l \in [A, B, \ldots, H]$: $\mathbf{f}^p_{i,l} = \bar{\mathbf{f}}^m_{i,j} + \bar{\mathbf{f}}^m_{i,j'}$, where the two movements $j, j' \in [1, 2, \ldots, 8]$. (2) Phase-pair representation: Once obtaining the phase embedding, a phase-pair representation is constructed to capture the pairwise relations for competition: $\mathbf{f}^{pp}_i = \mathbf{f}^p_{i,l} \| \mathbf{f}^p_{i,l'}$. (3) Phase competition: To avoid phase conflict, pairwise competition scores (Zheng et al., 2019) are learned as a competition mask, denoted as $\mathbf{\Omega}$. Thus, the phase-pair representations $\mathbf{f}^{pp}_i$ will go through 1×1 convolution and multiply with the phase competition mask to yield the masked phase-pair embedding: $\mathbf{f}^{pp}_{masked\ i} = Conv_{1\times1}(\mathbf{f}^{pp}_i) \otimes \mathbf{\Omega}$, and $\otimes$ is Kronecker product. We denote the whole FRAP module as $FRAP(\cdot)$:

$$\mathbf{f}^{pp}_{masked\ i} = FRAP(\bar{\mathbf{f}}^m_{i,j}) \tag{3}$$

Thirdly, the final policy $\pi(\cdot)$ and value $V(\cdot)$ estimation networks are implemented as the actor and critic, which are conducted by MLP, respectively. In contrast to the original FRAP which uses Deep Q Leaning (DQN) (Osband et al., 2016) as the action generator, the introduced actor-critic model can handle the action-continuous scenarios and is more friendly for the parallel training by A3C. To get the final action, the phase can be selected by the highest probability or sampled by probability distribution of the i-th intersection for exploration from $\pi(\mathbf{a}_i|f_i)$, where $\mathbf{a}_i \in \mathbb{R}^8$ for 8 phases.

$$Actor:\ \pi(\mathbf{a}_i|f_i) = Softmax\left(MLP\left(\sum(\mathbf{f}^{pp}_{actor\ i})\right)\right), \text{where } \mathbf{f}^{pp}_{actor\ i} = Conv_{1\times1}(\mathbf{f}^{pp}_{masked\ i}) \tag{4}$$

$$Critic:\ V(f_i) = MLP\left(\sum(\mathbf{f}^{pp}_{critic\ i})\right), \text{where } \mathbf{f}^{pp}_{critic\ i} = Conv_{1\times1}(\mathbf{f}^{pp}_{masked\ i}) \tag{5}$$

Overall, compared with the original FRAP, our main differences are three-fold: (1) A general plug-in module as well as the unified state and action are designed. Such a unified state space and action space enables the model to be co-trained with multi-scenarios and to handle several structure-distinct intersections simultaneously. (2) All eight movements are utilized as the state input at the same time, and each movement state is featured by traffic indexes and indicator embedding. In this way, the proposed framework gains better generalizability and various independent RL-based models can be transferred easily. (3) Actor and critic networks are adopted, which can be trained by A3C, thus each progress takes one scenario to interact with the model and returns the updated gradient.

4.3 Detailed RL Settings

State: To respect an actual situation, we measure each entering lane demand within 50m to the intersection (Chu et al., 2019). We choose five features from the observations as states: queue length, current phase, occupancy, flow, and the number of cars that are stopping (Wei et al., 2019c).
Action: The duration of each phase is fixed in our setting, so the action is to decide which phase to choose for the next based on the current states.
Reward: The reward is a weighted average of four reward components: queue length (L_i), wait time (T^w_i), delay time (T^d_i), and pressure (P_i), with detailed definitions and calculations in Appendix A. The reward R_t for the step t: $R_t = \sum_{c \in C} w_c c_i$, where w_c re-scales the value of each component c, C is the set of components $C = 4$, and c_i is the reward component c in intersection i at step t. The smaller the four reward components are, the better the traffic condition is. To maximize the total reward, the weights are set negative: $w_{delay\ time} = -1e^{-5}, w_{wait\ time} = -1e^{-3}, w_{queue\ length} = -1e^{-3}, w_{pressure} = -5e^{-3}$.

5 Experiment

5.1 Datasets

We choose the simulation of urban mobility (SUMO) (Behrisch et al., 2011) as the simulator. Seven different scenarios are employed for evaluation, and each one includes multiple intersections within a region. Four of them are publicly available (Ault & Sharon, 2021): *Grid 4×4*, *Arterial 4×4*, *Ingolstadt 21*, and *Cologne 8*. As another contribution for RL-based TSC, we build three scenarios, i.e, *Grid 5×5*, *Fenglin*, and *Nanshan*. We built *Fenglin* via the SUMO *netedit* module based on a real-world scenario and its traffic flow data is simulated based on the actual flow, more detail are written in Appendix C.2 . We also built *Nanshan* with 28 traffic lights via *OpenStreetMap* based on a real district in Shenzhen, China. During training, our model does not need manual labels. The details about the seven scenarios are introduced in Appendix C.

Table 1: Large scale co-training performance

Model	Rewards	*Grid 4×4*	*Arterial*	*Ingolstadt 21*	*Cologne 8*	*Fenglin*
FTC (single-scenario)	Total	-0.6376	-8.1955	-1.9223	-1.9565	-4.2271
	Delay time	662.6	6632.4	3066.5	2094.3	4250.4
	Wait time	280.6	1183.3	1129.8	1193.3	1852.9
	Queue length	205.8	2274.7	523.9	517.0	1512.5
	Pressure	329.0	1860.5	647.5	606.5	2901.0
MaxPressure (single-scenario)	Total	-0.4024	-7.9282	-1.1089	-0.6583	-1.7214
	Delay time	545.7	6032.1	2958.8	1353.8	2849.2
	Wait time	173.7	1150.8	742.0	397.8	808.6
	Queue length	116.4	1857.2	221.3	171.3	548.2
	Pressure	248.0	2264.0	302.5	199.5	1095.5
MPLight (single-scenario)	Total	-0.4385	-4.0535	-1.1628	-0.7669	-1.8667
	Delay time	561.0	4088.9	2808.4	1481.9	3057.7
	Wait time	187.9	704.4	770.5	455.6	791.6
	Queue length	129.8	1195.0	254.5	211.2	668.3
	Pressure	268.0	2082.0	329.5	224.5	1206.5
MetaLight (multi-scenario)	Total	-0.3745	**-3.7568**	-0.8130	-0.5517	-1.7336
	Delay time	537.5	3725.1	2937.4	1298.2	2775.2
	Wait time	163.1	544.7	568.7	330.0	818.6
	Queue length	104.8	1027.0	128.8	141.2	549.2
	Pressure	235.5	**1549.5**	225.0	172.0	1108.5
GESA-single (single-scenario)	Total	**-0.2643**	-6.5587	-0.7576	-0.5503	-2.8688
	Delay time	**479.0**	4158.8	2221.4	1292.6	3181.8
	Wait time	**111.8**	**326.1**	542.0	329.9	**641.9**
	Queue length	**69.4**	1285.1	118.3	140.2	954.6
	Pressure	**168.5**	2135.0	213.5	167.0	2225.0
GESA (multi-scenario co-train)	Total	-0.2761	-3.8856	**-0.7401**	**-0.4834**	**-1.5623**
	Delay time	487.9	**3552.6**	**2200.6**	**1239.1**	**2672.2**
	Wait time	117.1	517.1	**532.6**	**290.3**	742.9
	Queue length	73.5	**975.0**	**115.3**	**120.0**	**486.8**
	Pressure	174.0	1608.0	**199.0**	**152.5**	**1033.5**
Improvement	Total	35.64%	-3.43%	9.85%	14.13%	10.18%

5.2 Benchmark Methods

Several representative peers are selected, including both traditional methods and RL-based methods.

- **Fixed-timed control (FTC)** (Roess et al., 2004): A fixed cycle time and phase order are set manually for each traffic light according to expert knowledge.
- **MaxPressure** (Varaiya, 2013; Kouvelas et al., 2014): As a popular choice in traditional TSC, it sets the phase with the max pressure between phases as green according to the queue length.
- **MPLight** (Chen et al., 2020): It is an extension work based on FRAP. It improves FRAP by utilizing pressure as both the state and reward for a DQN agent.
- **MetaLight** (Zang et al., 2020): It combines Meta RL with FRAP. However, it needs heavy labelling work, and the training scenarios are quite similar.
- **GESA-single**: We also test our GESA's performance in a single scenario training setting.
- **GESA** (The proposed model): It is the only method that can be co-trained in multiple scenarios.

5.3 Evaluation on Large-scale scenarios co-training

The GESA is co-trained with all seven scenarios. The learning curves in each scenario are illustrated in Fig. 5(a). We observe that training on *Grid 4×4* is the most stable with the highest reward due to its simple topology. The training variance on *Nanshan* and *Fenglin* scenarios are the largest since their settings are realistic and complex. Compared with GESA trained in the single scenario in Fig. 5.(b), co-training can also converge faster and more stable.

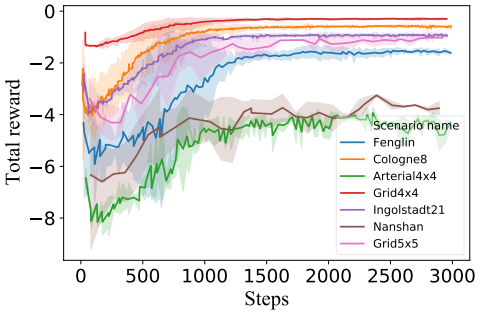
The numerical results of all methods in five scenarios are summarized in Table 1, with the best results in boldface and the second-best underlined. Compared to the benchmarks, our GESA achieves the highest total reward over the four scenarios and the second-highest in the *Arterial 4×4*. On average, we achieve **+13.27%** improvement compared to the best benchmarks and **+14.38%** than MetaLight.

Table 2: Generalization Power of GESA: GESA Transfer *v.s.* Baseline's Single-scenario Training

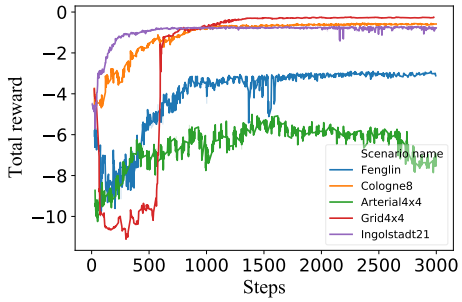
Model	Rewards	Unseen Scenario for GESA				
		Grid 4×4	*Arterial 4×4*	*Ingolstadt 21*	*Cologne 8*	*Fenglin*
Best of the baselines	Total	-0.3745	-3.7568	-0.8130	-0.5517	-1.7214
GESA	Total	**-0.3085**	**-3.6940**	**-0.7706**	**-0.4720**	**-1.6964**
	Delay time	510.4	3888.4	2252.5	1238.6	2634.0
	Wait time	132.4	595.5	564.2	281.0	808.9
	Queue length	84.40	1108.2	114.4	118.3	535.6
	Pressure	189.0	1825.5	194.5	152.5	1145.0
Improvement	Total	21.39%	1.70%	5.50%	16.89%	1.47%

In the *Arterial 4×4* , we achieve the best sub-rewards in queue length and delay time. This may be because for some scenarios, there are conflicts between the sub-rewards, making it difficult for GESA to promote them in all the scenarios simultaneously. Thus, MetaLight took the advantage of only training two scenarios each time. However, this comes with a curse that MetaLight's training is quite unstable duo to the heterogeneity of our scenarios, as shown in Appx. D.2, thus it cannot beat GESA in all other scenarios.

In the *Grid 4×4*, the GESA-single model obtains higher rewards than the full GESA since this scenario is straightforward with low traffic volume. However, in more complex scenarios, GESA-single performs worse since it tends to get stuck in a local optimum. The full GESA instead has seen more diverse scenarios. A detailed explanation is given in Appx. E.3.2.



(a) Co-training in multi-scenarios



(b) Training in single scenario

Figure 5: Learning curves in all scenarios

5.4 Dealing with a new scenario

To study the transferability of the model, we perform the leave-one-out operation: namely, training the model on six scenarios and testing in the seventh scenario that is unseen.We rotate the unseen scenario among the five scenarios in turn, as shown in Table 2, where each column shows the evaluated results on the taken-out (unseen) scenario. The performance from the second-best benchmark is also listed. Our framework still achieves a 9.39% average improvement over the five scenarios, which demonstrates our model's higher generalizability.

5.5 More Experiments: Ablation Studies

We also conducted ablation studies: (1) Appx. E.1 shows the necesseity of training simultaneously; (2) Ablation on rewards (Appx. E.2) shows four rewards being trained together is necessary to achieve better performance than single-reward versions; (3) Ablation on training different number of scenarios (Appx. E.3) shows the steadily increase of performance with more co-trained scenarios.

6 Conclusions

In this paper, to narrow the *sim2real* gap, we propose GESA, a general RL-based framework for the traffic signal control task, which can obtain a general model through large-scale training with multiple scenarios. We first present a general plug-in module to eschew manual annotation in the structure-varying intersections. To keep the model input and output consistent under different scenarios, we elaborately design the unified state space and action space. Based on these unification operations, we modify the original FRAP to a policy gradient-based framework, which facilitates the model coverage and is compatible with different intersections. Experiments show that our framework outperforms the benchmarks, even if the evaluated scenario is unseen in the training stage.

REFERENCES

James Ault and Guni Sharon. Reinforcement learning benchmarks for traffic signal control. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

Michael Behrisch, Laura Bieker, Jakob Erdmann, and Daniel Krajzewicz. SUMO–Simulation of Urban MObility: An overview. In *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind, 2011.

Noe Casas. Deep deterministic policy gradient for urban traffic light control. *arXiv preprint arXiv:1703.09035*, 2017.

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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3414–3421, 2020.

Tianshu Chu, Jie Wang, Lara Codecà, and Zhaojian Li. Multi-agent deep reinforcement learning for large-scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 21(3): 1086–1095, 2019.

Samah El-Tantawy, Baher Abdulhai, and Hossam Abdelgawad. Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown Toronto. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1140–1150, 2013.

Martin Fellendorf. VISSIM: A microscopic simulation tool to evaluate actuated signal control including bus priority. In *64th Institute of Transportation Engineers Annual Meeting*, volume 32, pp. 1–9. Springer, 1994.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Juntao Gao, Yulong Shen, Jia Liu, Minoru Ito, and Norio Shiratori. Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network. *arXiv preprint arXiv:1705.02755*, 2017.

PB Hunt, DI Robertson, RD Bretherton, and M Cr Royle. The SCOOT on-line traffic signal optimisation technique. *Traffic Engineering & Control*, 23(4), 1982.

Peter Koonce and Lee Rodegerdts. Traffic signal timing manual. Technical report, United States. Federal Highway Administration, 2008.

Anastasios Kouvelas, Jennie Lioris, S Alireza Fayazi, and Pravin Varaiya. Maximum pressure controller for stabilizing queues in signalized arterial networks. *Transportation Research Record*, 2421(1):133–141, 2014.

Silas C Lobo, Stefan Neumeier, Evelio MG Fernandez, and Christian Facchi. Intas–the ingolstadt traffic scenario for sumo. *arXiv preprint arXiv:2011.11995*, 2020.

PR Lowrie. SCATS, Sydney co-ordinated adaptive traffic system: A traffic responsive method of controlling urban traffic. 1990.

Jinming Ma and Feng Wu. Feudal multi-agent deep reinforcement learning for traffic signal control. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 816–824, 2020.

Pitu Mirchandani and Larry Head. A real-time traffic signal control system: architecture, algorithms, and analysis. *Transportation Research Part C: Emerging Technologies*, 9(6):415–432, 2001.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937. PMLR, 2016.

Tomoki Nishi, Keisuke Otaki, Keiichiro Hayakawa, and Takayoshi Yoshimura. Traffic signal control based on reinforcement learning with graph convolutional neural nets. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 877–883. IEEE, 2018.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped DQN. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 4033–4041, 2016.

LA Prashanth and Shalabh Bhatnagar. Reinforcement learning with average cost for adaptive control of traffic lights at intersections. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 1640–1645. IEEE, 2011.

Roger P Roess, Elena S Prassas, and William R McShane. *Traffic engineering*. Pearson/Prentice Hall, 2004.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Merlijn Steingrover, Roelant Schouten, Stefan Peelen, Emil Nijhuis, Bram Bakker, et al. Reinforcement learning of traffic light controllers adapting to traffic congestion. In *BNAIC*, pp. 216–223, 2005.

Elise Van der Pol and Frans A Oliehoek. Coordinated deep reinforcement learners for traffic light control. *Proceedings of Learning, Inference and Control of Multi-agent Systems (at NIPS 2016)*, 1, 2016.

Pravin Varaiya. The max-pressure controller for arbitrary networks of signalized intersections. In *Advances in Dynamic Network Modeling in Complex Transportation Systems*, pp. 27–66. Springer, 2013.

Christian Varschen and Peter Wagner. Mikroskopische modellierung der personenverkehrsnachfrage auf basis von zeitverwendungstagebüchern. *Integrierte Mikro-Simulation von Raum-und Verkehrsentwicklung. Theorie, Konzepte, Modelle, Praxis*, 81:63–69, 2006.

Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. Intellilight: A reinforcement learning approach for intelligent traffic light control. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2496–2505, 2018.

Hua Wei, Chacha Chen, Guanjie Zheng, Kan Wu, Vikash Gayah, Kai Xu, and Zhenhui Li. Presslight: Learning max pressure control to coordinate traffic signals in arterial network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1290–1298, 2019a.

Hua Wei, Nan Xu, Huichu Zhang, Guanjie Zheng, Xinshi Zang, Chacha Chen, Weinan Zhang, Yanmin Zhu, Kai Xu, and Zhenhui Li. Colight: Learning network-level cooperation for traffic signal control. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1913–1922, 2019b.

Hua Wei, Guanjie Zheng, Vikash Gayah, and Zhenhui Li. A survey on traffic signal control methods. *arXiv preprint arXiv:1904.08117*, 2019c.

Lun-Hui Xu, Xin-Hai Xia, and Qiang Luo. The study of reinforcement learning for traffic self-adaptive control under multiagent markov game environment. *Mathematical Problems in Engineering*, 2013, 2013.

Kok-Lim Alvin Yau, Junaid Qadir, Hooi Ling Khoo, Mee Hong Ling, and Peter Komisarczuk. A survey on reinforcement learning models and algorithms for traffic signal control. *ACM Computing Surveys (CSUR)*, 50(3):1–38, 2017.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

Xinshi Zang, Huaxiu Yao, Guanjie Zheng, Nan Xu, Kai Xu, and Zhenhui Li. Metalight: Value-based meta-reinforcement learning for traffic signal control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 1153–1160, 2020.

Zhi Zhang, Jiachen Yang, and Hongyuan Zha. Integrating independent and centralized multi-agent reinforcement learning for traffic signal network optimization. pp. 2083–2085, 2020.

Guanjie Zheng, Yuanhao Xiong, Xinshi Zang, Jie Feng, Hua Wei, Huichu Zhang, Yong Li, Kai Xu, and Zhenhui Li. Learning phase competition for traffic signal control. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1963–1972, 2019.

Liwen Zhu, Peixi Peng, Zongqing Lu, and Yonghong Tian. MTLight: Efficient multi-task reinforcement learning for traffic signal control. In *ICLR 2022 Workshop on Gamification and Multiagent Solutions*, 2022.

APPENDIX

The appendix will give experimental details and further results. Appendix A will define the rewards in detail; Appendix B will give the parameter setting; Appendix C will introduce the seven scenarios with more details. Specifically, we will further introduce the two scenarios constructed by ourselves, i.e., *Fenglin* and *Nanshan*, and we will also further introduce how the traffic flow is simulated based on the actual flow. Appendix D.2 will show the training process of MetaLight. Appendix E will show the ablation studies, with ablations on the model components in Appendix E.1, on reward in Appendix E.2, and on the number of co-trained scenarios in Appendix E.3. The Appendix F will give some visualization on the signal controlling. The appendix G will discuss about dealing the intersections with more than four entering approaches and future work.

A REWARD SPECIFICATIONS

As shown in Section 4.3, the weighted combination of four components is adopted as the reward function. Their definitions are listed as follows.

Delay Time T_i^d***:*** *The delay time for the intersection* i *is the average of all the lanes' delay time. The delay time for a lane is the sum of all vehicles' delay time on the lane. A vehicle delay time* d_v *is calculated by:*

$$d_v = \Delta t \left(1 - \frac{v_v(t)}{v_{max}}\right), \tag{6}$$

where Δt *is interval time,* $v_v(t)$ *is the average speed over the interval time, and* v_{max} *is the max speed of vehicle v. Its unit is second (s).*

Wait Time T_i^w***:*** *The wait time for the intersection* i *is the average of all the lanes it contains, whereas the wait time for a particular lane is the sum of all-vehicles stop time on it. The unit is second* (s).

Queue Length L_i***:*** *The queue length is defined as the average of all-lanes vehicle queue length in an intersection, which is in meter* (m).

Pressure P_i***:*** *As it is first introduced in (Chen et al., 2020) and as shown in Fig. 6, the pressure of the intersection* i *is the difference between the queuing vehicles numbers on entering and exiting approaches. Intuitively, a higher pressure indicates a greater imbalance between the entering and exiting lanes.*

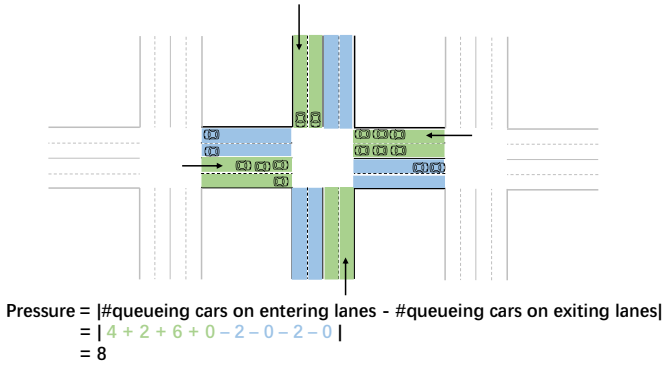


Figure 6: The illustration of pressure.

B Parameters Setting

B.1 Reward Weights Setting

As described in section 4.3, the re-scale weights w_* are introduced to balance each component in the reward. In our experimental, we decided the weights according to parameter tuning so that the four reward components are in the similar scale. Without balancing the four components, the training will be led to one direction that only maximizes the most dominant reward component. As mentioned before, we set $w_{queue\ length} = -1e^{-3}$, $w_{wait\ time} = -1e^{-3}$, $w_{delay\ time} = -1e^{-5}$, $w_{pressure} = -5e^{-3}$.

B.2 Training Parameters Setting

The total simulation steps of each scenario are 3600*s*, and the agent interacts with the simulation environment every 15 seconds. During the testing stage, we repeatedly evaluate the method five times under different random seeds, and the average is regarded as the final result. For GESA, the learning rate is set as 10^{-4}, the batch size is 128, and the rollout length is set to 128. Generalized advantage estimation (Schulman et al., 2015) is adopted as the value estimation method. After each rollout and return of the gradient, the shared model updates parameters according to the newest parameter in each process of A3C (Mnih et al., 2016).

C The Details of Scenarios for Training

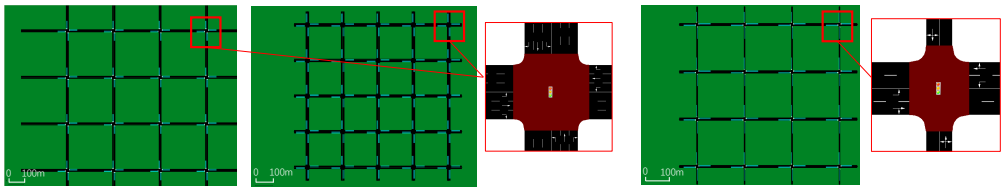
In this section, the seven scenarios will be introduced in detail. Some key specifications of these scenarios are summarized in the Table 3.

Table 3: Scenarios for Training

Scenario	Total # of signaled intersections (Int.)	# of 2-way Int.	# of 3-way Int.	# of 4-way Int.
Grid 4×4	16	0	0	16
Arterial 4×4	16	0	0	16
Ingolstadt 21	21	0	17	4
Cologne 8	8	1	3	4
Grid 5×5	25	0	0	25
Fenglin	7	0	2	5
Nanshan	28	1	6	21

C.1 Open-source Scenarios

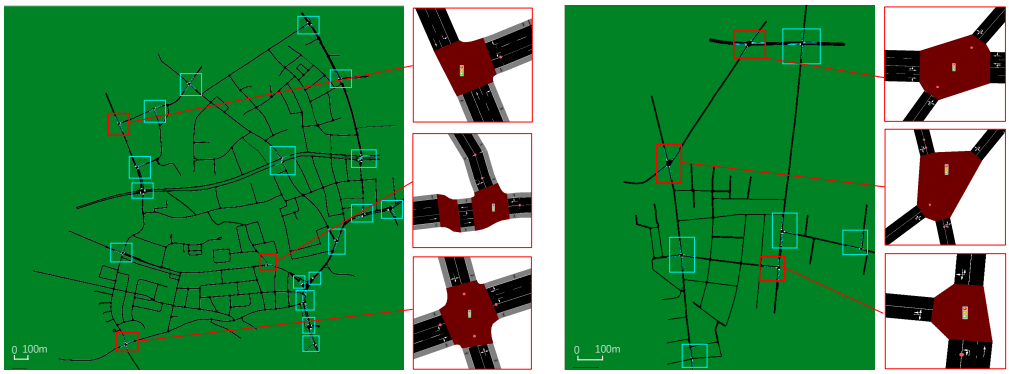
The details for *Grid 4×4* (Chen et al., 2020), *Grid 5×5*, and *Arterial 4×4* (Ma & Wu, 2020) are given in Figure 7. The *Grid 5×5* is designed by ourselves, following the same setting of *Grid 4×4*.



(a) *Grid 4×4* (b) *Grid 5×5* (c) *Arterial 4×4*

Figure 7: The scenarios of (a) *Grid 4×4*, (b) *Grid 5×5*, with all the intersections signaled and each entering approach having three lanes with movements of left, through, and right, respectively; (c) *Arterial 4×4*, with all the intersections signaled, E & W entering approaches having two lanes with movements of left and right-through, as N & W entering approaches having one lane with the movement of left-through-right. (The blue strips indicate the locations of 50-meter detectors.)

The scenarios *Ingolstadt 21* (Lobo et al., 2020) [1] and *Cologne 8* (Varschen & Wagner, 2006) [2] describe the one-day traffic within the city of Ingolstadt and Cologne in Germany, respectively, as shown in Figure. 8.



(a) *Ingolstadt 21* (b) *Cologne 8*

Figure 8: The scenarios of (a) *Ingolstadt 21*, (b) *Cologne 8*, with the signaled intersections highlighted and three intersections zoomed for demonstration.

C.2 Two Real-world Scenarios

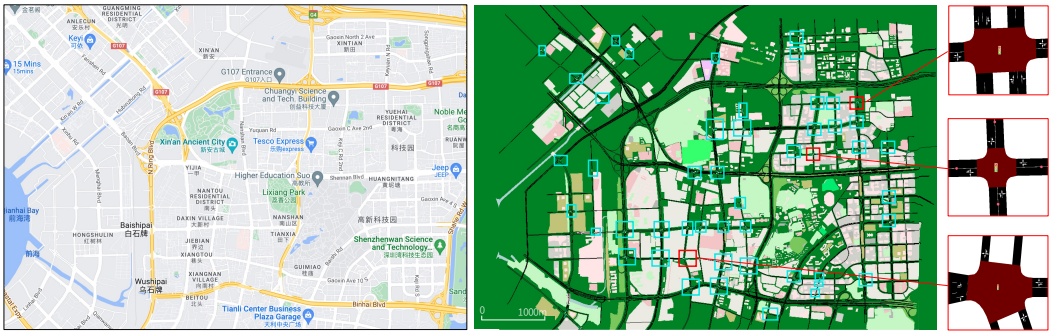
To build more realistic scenarios, we manually construct the *Nanshan* scenario based on Nanshan district in Shenzhen, China and *Fenglin* scenario based on Fenglin corridor in Shaoxin, China.

For the *Nanshan* scenario, we generate the road network based on OpenStreetMap, and we set the flow pattern and the initial phase setting as the SUMO default.

For *Fenglin* scenario, it is even more elaborately designed such that it is as real as possible. As illustrated in Fig. 10.(a), it is a corridor network in Shaoxing, China, and there are seven intersections that are with signal controlling. There are mainly four steps when constructing this scenario:

- Constructing the road network: We built the road network via the SUMO *netedit*[3] module based on the Fenglin West Road in Shaoxin;
- Setting the movements for each entering approach: For example the north entering approach has two through lanes, two left lanes, and one through-right lane;
- Setting the traffic signal initial phase: we use the default setting;

[1] https://sumo.dlr.de/docs/Data/Scenarios/TAPASCologne.html
[2] https://github.com/silaslobo/InTAS
[3] https://sumo.dlr.de/docs/Netedit/index.html

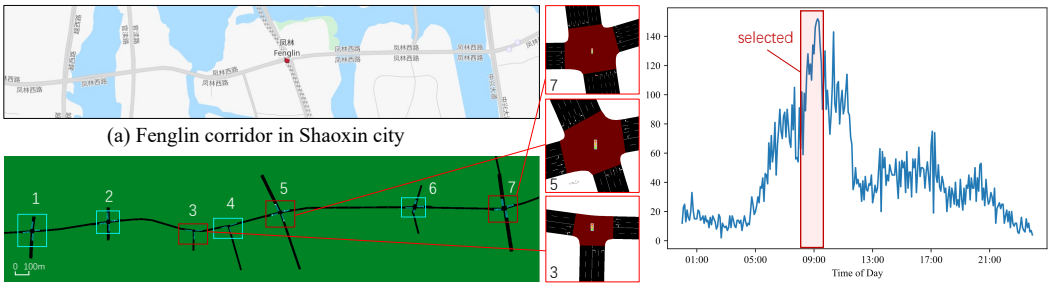


(a) Nanshan District in Shenzhen (b) *Nanshan* scenario in SUMO

Figure 9: (a) Nanshan district map, (b) *Nanshan* scenario in SUMO, with the signaled intersections highlighted and three intersections zoomed for demonstration.

- Setting the traffic flow: For each intersection, live cameras are deployed to count the passing vehicles and calculate the average traffic flow, as shown in Fig. 10.(c). (1) We select the peak hour in the red box, and inject the flow from the boundary intersection, e.g., intersection 1, into the network; (2) The flow split ratio also needs to be defined. We calculate it based on the real data for each entering approach. For example, the ratio of through:right:left for intersection 5 is 5:3:2; (3) Lastly, we place 50-meter Lane-Area (E2) detectors [4] in each entering lane to get the observations such as queue length and occupancy, and we place Multi-Entry-Exit Detectors (E3) [5] between the entry and exit to get the delay time and wait time.

After the aforementioned designs, the *Fenglin* scenario reflects the reality very well.



(a) Fenglin corridor in Shaoxin city

(b) *Fenglin* scenario in SUMO (c) Flow pattern in the Intersection-2

Figure 10: (a) Fenglin corridor map; (b) *Fenglin* scenario in SUMO, with the signaled intersections highlighted and three intersections zoomed for demonstration; and (c) Flow pattern at the intersection-2 during the morning peak hour around 9 AM.

D Additional Results for GESA and MetaLight

D.1 The Reuslts of GESA Compared with Other Benchmarks

The Table 1 can also be presented in a plot format for better comparison. In the Fig.11, each reward is normalized by dividing the maximal value.

[4] https://sumo.dlr.de/docs/Simulation/Output/Lanearea_Detectors_%28E2%29.html

[5] https://sumo.dlr.de/docs/Simulation/Output/Multi-Entry-Exit_Detectors_%28E3%29.html

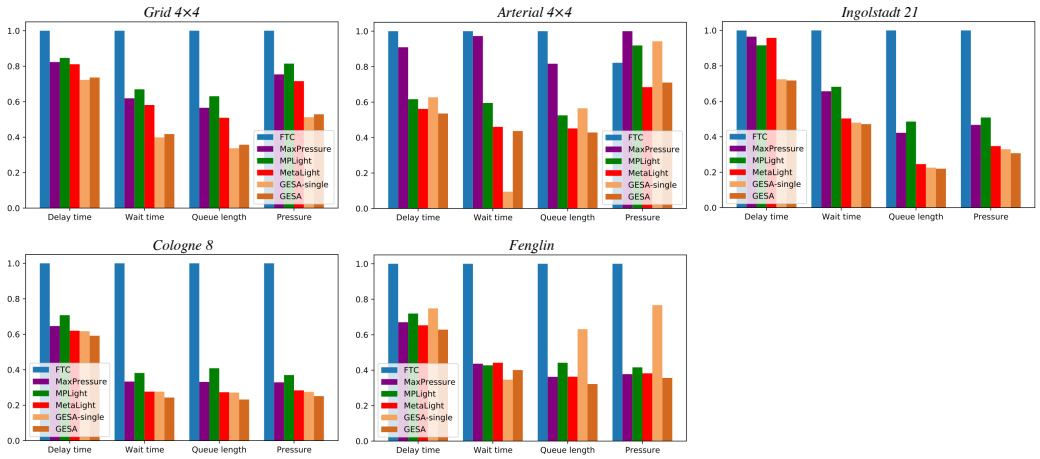
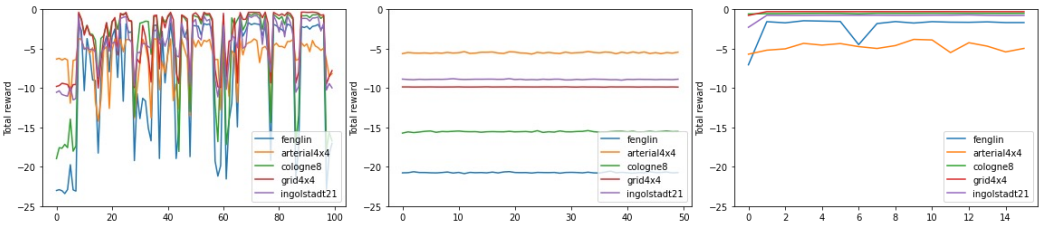


Figure 11: Performance comparison of Four Reward components in the five different scenarios.

D.2 Results for MetaLight

As shown in Fig.12.(a), we display the total reward of MetaLight in each checkpoint during evaluating. Since the Meta-Learner in MetaLight is trained with only two scenarios observed each time, and our scenarios are quite heterogenous to each other, so the evaluating results are quite unstable, with the total rewards fluctuating largely. Our GESA instead is improving total reward more steadily, as shown in Fig.5.(a) and Fig.12.(c).



(a) Evaluation of MetaLight (b) Evaluation of GESA-sequential-train (c) Evaluation of GESA

Figure 12: (a) Evaluation of MetaLight; (b) Evaluation of GESA with the Actor-Critic feeding the scenarios in a sequential manner; (c) Evaluation of GESA.

E Ablation Studies

In this section, detailed ablation studies will be conducted to further demonstrate the necessity of our model design. Section E.1 will compare the effect of training A3C in a sequential manner. Section E.2 studies the effects of each reward. Section E.3 studies how introducing different amount of scenarios for co-training will affect the performance.

E.1 Ablation Study on the Model Components

The proposed plug-in module is an automated labeling module, which does not affect the performance of the model, so its ablation is not available.

Alternatively, to validate the importance of the A3C module, we train the GESA model in a sequential manner (i.e., feeding the scenarios one by one to train the model) rather than the original parallel manner (i.e., training model with all the scenarios simultaneously). This variant is termed as GESA-sequential-training, and its training curves are shown in Fig.12.(b). The rewards of all the models are small and without upward trend, which means that they do not converge towards favorable outcomes and the sequential manner is not applicable to the training. Compared to it, the

Table 4: single-component vs 4-component reward

Component	Rewards	*Grid 4x4*	*Arterial 4×4*	*Ingolstadt 21*	*Cologne 8*	*Fenglin*
$\text{GESA}_{\text{delay time}}$	Total	-0.3266	-4.8884	-0.8715	-0.5847	-1.5905
	Delay time	511.7	3906.4	2267.1	1266.1	2855.5
	Wait time	139.4	420.8	594.2	346.3	**563.6**
	Queue length	89.9	1137.7	154.7	154.3	550.2
	Pressure	210.0	1960.5	276.5	188.0	1395.0
$\text{GESA}_{\text{wait time}}$	Total	-10.1055	-4.8713	-7.1326	-11.1791	-16.4658
	Delay time	5642.4	2607.8	5513.5	7698.9	11995.5
	Wait time	655.2	**189.3**	**516.1**	674.6	977.4
	Queue length	2907.6	985.4	1870.6	3201.8	3332.4
	Pressure	5121.0	2178.0	3337.5	3369.5	14766.0
$\text{GESA}_{\text{queue length}}$	Total	-0.3005	-5.4331	-0.9334	-0.5207	-1.6324
	Delay time	516.0	4071.8	2258.4	1291.0	2636.7
	Wait time	127.4	515.6	624.4	315.8	749.2
	Queue length	82.60	1175.6	182.5	126.3	518.7
	Pressure	186.5	1850.5	281.0	167.0	1149.0
$\text{GESA}_{\text{pressure}}$	Total	-0.3261	-6.3235	-1.2017	-0.5417	-1.5831
	Delay time	512.4	4306.6	2637.2	1334.8	**2598.8**
	Wait time	138.4	607.1	792.2	326.6	721.5
	Queue length	91.0	1386.4	228.1	132.7	517.2
	Pressure	204.5	1900.0	354.5	172.5	1107.5
GESA	Total	**-0.2761**	**-3.8856**	**-0.7401**	**-0.4834**	**-1.5623**
	Delay time	**487.9**	**3552.6**	**2200.6**	**1239.1**	2672.2
	Wait time	**117.1**	517.1	532.6	**290.3**	742.9
	Queue length	**73.5**	**975.0**	**115.3**	**120.0**	**486.8**
	Pressure	**174.0**	**1608.0**	**199.0**	**152.5**	**1033.5**

parallel manner (implemented by A3C) can significantly improve the rewards (Fig.12.(c)), which contributes to the training.

E.2 Ablation Study on Rewards

Since the designed reward functions contains four components (that is, queue length, wait time, delay time, and pressure), we perform the component ablation experiments to provide insights into their impact on the model performance specifically.

We conduct a single-component v.s. all-component reward ablation experiment in the co-training setting, the results are shown in Table 4. Grey values represent that the corresponding components are excluded in the reward function during training but measured in the evaluation. The best performance is in boldface and the near-best one is underlined. It can be seen that even though the GESA trained with single component may outperforms under the particular metric, its performance on the remaining metrics is unremarkable. For example, in the scenario *Ingolstadt 21*, $\text{GESA}_{\text{wait time}}$ achieves the best on the wait time metric, but degrades about 150.5%, 1522.4%, 1577.1% on the delay time, queue length, and pressure compared to the canonical GESA, respectively. The higher delay time and lower wait time indicate that the vehicles move slower even though stop infrequently, and there is a higher potential for jams accordingly.

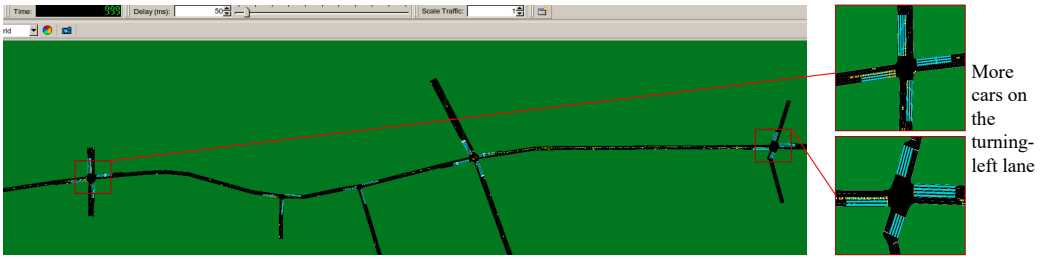
Overall, given the intrinsic correlations in the four components, it is easy to fall into sub-optimum when only optimizing one specific component. The canonical GESA with four-component reward achieves promising results in most cases. The highest total reward is more straightforward to demonstrate its superiority.

E.3 Ablation Study on the Number of Co-trained Scenarios

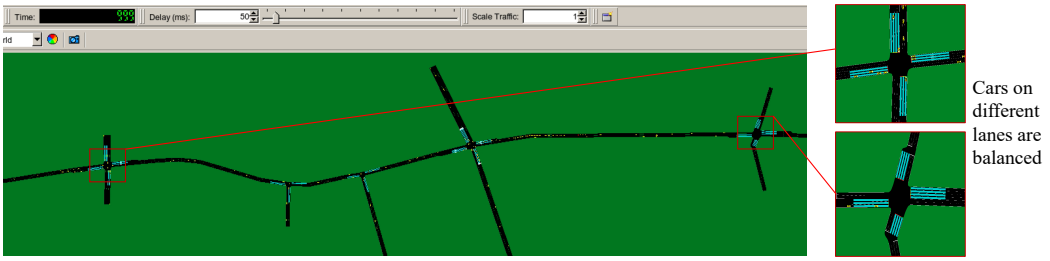
This subsection will exam the effect of co-training different amount of scenarios on the model performance in E.3.1, as well as explain in detail why multi-scenario co-training is better than single-scenario training in E.3.2.

Table 5: Results of joint training with different numbers of scenarios based on *Arterial 4×4* scenario

The number of scenarios	Chosen scenarios	Reward of *Arterial 4×4*	Improvement
1	***Arterial 4×4***	-6.5587	-
3	***Arterial 4×4*** + *Cologne 8*, *Nanshan*	-6.3182	3.67%
5	***Arterial 4×4*** + *Cologne 8*, *Nanshan* + *Grid 4×4*, *Ingolstadt 21*	-4.3198	34.14%
7	***Arterial 4×4*** + *Cologne 8*, *Nanshan* + *Grid 4×4*, *Ingolstadt 21* + *Grid5x5*, *Fenglin*	-3.8856	40.76%



(a) GESA-single visualization of *Fenglin*



(b) GESA visualization of *Fenglin*

Figure 13: Compare single training and co-training visualization result in *Fenglin* scenario

E.3.1 Performance Increases with More Scenarios

We use *Arterial 4×4* as the anchor to verify the impact of the number of co-trained scenarios, because there is a big gap of its results in single-scenario v.s. multi-scenario. We increase the co-trained scenarios with two stride lengths, i.e., single, three, five, and seven, and then perform the test, respectively. The result is shown in Table 5. It can be observed that the reward increases monotonically as the number of co-training scenarios rising. This means that adding more scenarios can facilitate the model performance, and the proposed general scenario-agnostic method works very well. This may be because the different scenarios may share similar latent spatiotemporal correlation, and co-training achieve the effect of "$1 + 1 > 2$" by leveraging this spatiotemporal correlation.

E.3.2 Why Multiple Scenarios Outperforms Single Scenario

As demostrated in Table 1 and Table 5, the multi-scenario co-training are much better than the single-scenario in most cases. Through visualization, we find that single-scenario training is prone to falling into local optima than large scale co-training.

A toy example is *Fenglin*. In Figure 13, we plot the traffic status in a particular simulation step in this scenario. One can find that the traffic controlled by the co-training model is less congested than the single-training model (the model only trained by *Fenglin* scenario) at the same time step. A obvious issue of the single-training one is that its left-turn traffic movement has a low priority of release, resulting more queue length and pressure in the left lanes.

This phenomenon is interpretable: the *Fenglin* road network is a corridor, which has an East-West main road and multiple North-South branch roads. Vehicles spend much more time on main road than on branch roads, therefore, it is easier for the agent to learn that the priority of letting go straight is higher than turning left. Once the agent is over-optimized in taking "go straight" action, it is difficult to break out, because it will get smaller rewards for exploration in a long time. This situation will not be encountered in the multi-scenario co-training model, since it handles diverse scenarios with various road network and traffic flows, and an agent with better generalization can be learned.

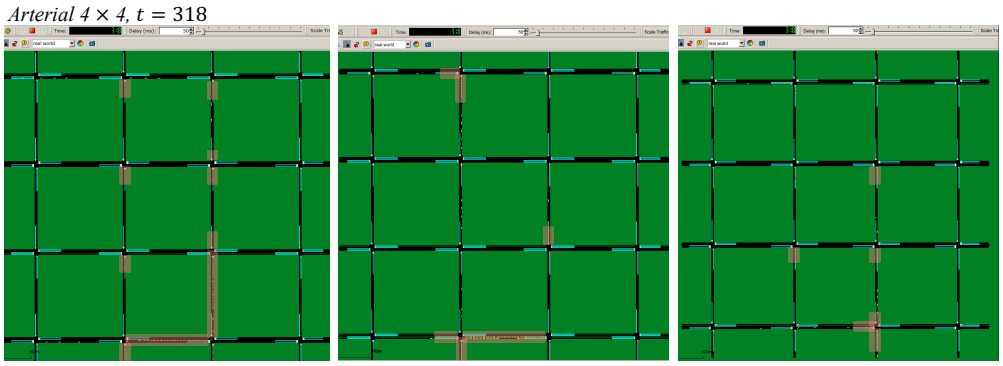
This reason can also explain why the improvement of multi-scenario performance is trivial compared to the one in *Grid 4×4*, the single scenario. The road network of *Grid 4×4* scenario is symmetrical, and the distribution of traffic flow is smooth and relatively uniform. While in *Arterial 4×4*, even if its road network is very symmetrical, its traffic flow distribution is very uneven, the top and tail row intersections have heavy traffic flow and two middle row intersections have no traffic flow in East-West lanes.

F Demo Visualizations

In this section, we will demonstrate the effectiveness of our model for real-time traffic signal control across three comparing models on three selected scenarios. The real-time visualization is available in the anonymous link `https://github.com/AnonymousIDforSubmission/GESA`, and we also offer some snapshots for comparison on each scenario at the same timestamp.

The traffic status during simulating for *Arterial 4×4* and *Fenglin* are illustrated in Figures 14 and 15, respectively. The congestion queue is highlighted in red. However, we would like to emphasize that in the demo, the human eyes could only tell the difference of queue length visually. Yet the whole traffic situation is evaluated by four rewards from different perspectives. And the metrics such as delay time and wait time are not easy to be perceived visually.

Fig. 14 shows the 318-th simulation step in the *Arterial 4×4* scenario. Due to the high traffic volume, longer queues are formed in several intersections when the traffic are controlled by the FTC method (Fig. 14.(a)). The GESA-Single model (Fig. 14.(b)) can alleviate the condition to some extent, while our GESA model (Fig. 14.(c)) shortens the queue length significantly, which demonstrates its superiority intuitively.



(a) FTC (b) GESA-Single (c) GESA

Figure 14: Comparison in *Arterial 4×4* with the methods (a) FTC, (b) GESA-Single, and (c) GESA.

Similarly, we also list the snapshots in Fig. 15 in *Fenglin* scenario where the traffic flow is generated based on the real-world peak hour. The long queues in the central intersection are also formed

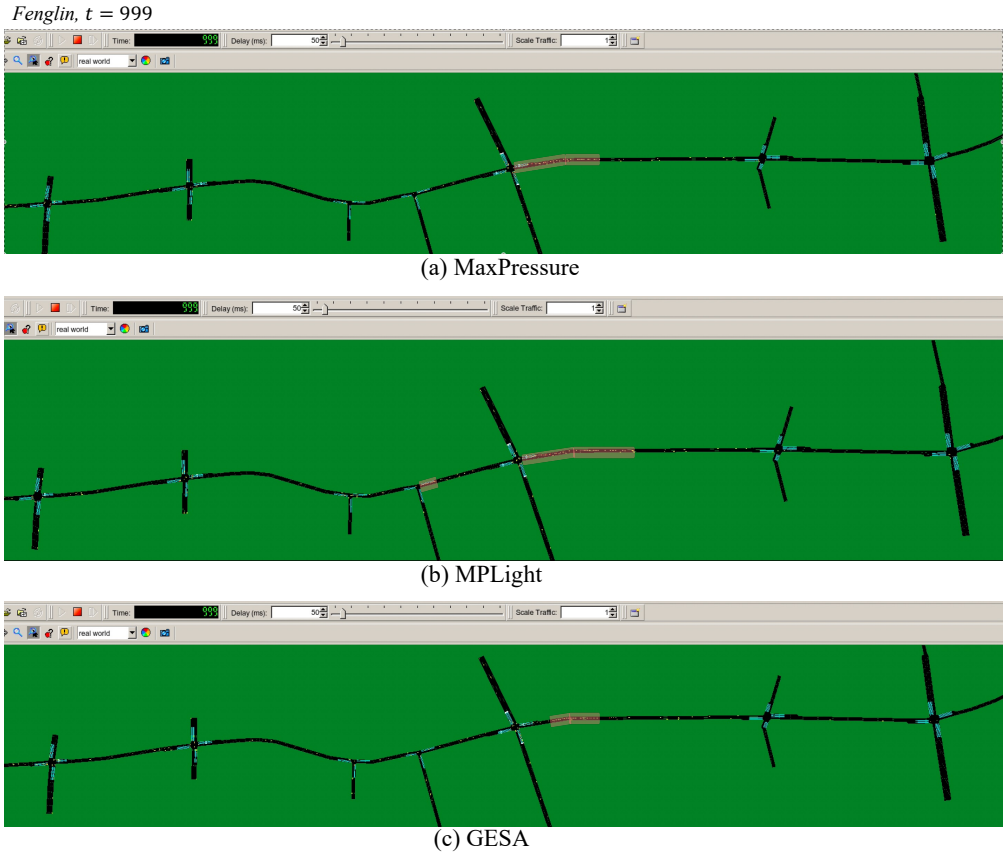


Figure 15: Comparison in *Fenglin* with the methods (a) MaxPressure, (b) MPLight, and (c) GESA.

when conducting MaxPressure (Fig. 15.(a)) and MPLight (Fig. 15.(b)), while our GESA model still remains a shorter queue.

G Discussion, Future Work, and Appreciation

In this section, we will provide additional discussions, future work, and our appreciation.

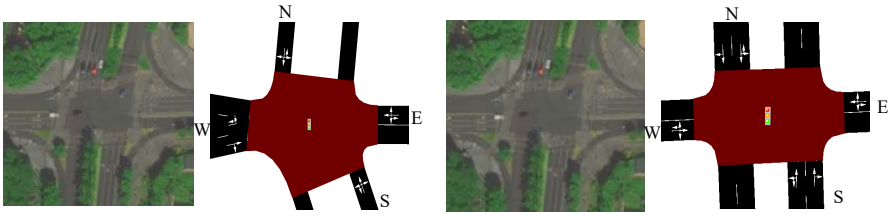
G.1 Discussion about Intersections with More Than Four Approaches

By the proposed GPI module (as shown in Algorithm 1), the intersections with no more than four approaches can be handled. It should be noted that it is a demonstration and the threshold "four" is not fixed, that is, intersections with five, six, or even more approaches can also be treated following the similar pipeline. When the threshold is enlarged, one need to (1) modify the angles interval in Algorithm 1, (2) update the conflict matrix of movements (in Fig. 2.(b)) and available phases, and (3) enlarge the model output (i.e., action space).

The reasons why we choose "four" are: (1) For the intersections in real-world road networks, two-approach, three-approach, and four-approach ones occupy dominant proportions[6]. (2) According to our practical statistics in Nanshan and Fenglin districts, there are no intersections with more than four entering approaches, so setting "four" in the proposed GPI module is quite enough to handle the realistic scenarios. Overall, the > 4-approach intersections are relatively rare, and the design for such intersections in GPI module is not worth the cost. If needed, we will consider to extend the GPI module to handle such situations specifically in our future work.

[6] https://en.wikipedia.org/wiki/Intersection_(road)

Moreover, one may be confused with the terms legs and approaches for an intersection. For example, a six-legged intersection can still have only four entering approaches, as shown in Fig. 16. The rest two legs are exiting approaches without need for signal control.



(a) 6-legged Intersection Example 1 in *Nanshan* (b) 6-legged Intersection Example 2 in *Nanshan*

Figure 16: Examples of 6-legged intersections with only four entering approaches.

G.2 Future Work

Our model further narrows the gap between simulation and real-world implementation. In the future, we plan to implement the GESA model in the real cities. We will also consider the intersections with more than four entering approaches.

G.3 Appreciation

The authors would like to thank the anonymous reviewers for their invaluable insights.