Appendix

A Notations

Table A1. Notations Used in Meta-CSLight.

Notations	Descriptions
(l, n)	The traffic movement from lane l to lane n.
$m(l, n)$	The traffic pressure of (l, n).
P_i	The traffic pressure of intersection i.
$DIS_{i,j}$	The distance between two adjacent intersections i and j.
v	The average vehicle travel speed.
Δt	The offset of the green signals between two adjacent intersections in the same direction.
$\mathcal{S}, s_{e,j}^t$	$\mathcal{S}$ denotes the joint state space observed by all agents; $s_{e,j}^t$ denotes the state representation of agent j at time step t in traffic flow V_e.
$\mathcal{A}$	The joint action space of all agents.
$\mathcal{P}$	The transition function maps the state-action pair at time t to the state at time $t+1$.
$\mathcal{R}, r_i^t, ri_i^t, rs_i^t$	$\mathcal{R}$ is the reward function; r_i^t is the reward of agent i; ri_i^t is the immediate reward of agent i; rs_i^t is the reward shaping of agent i.
γ	Discount factor.
V_e, W_e	The traffic flow environments, where e is the index of traffic flow environments.
$f_{\theta_e^{\text{indep}}}$	The independent learning model.
$f_{\theta_e^{\text{nei}}}$	The neighbor-aware learning model.
$q_{e,j}^t$	The predicted Q-value vector of agent j.
$\mathcal{N}_k$	The neighbor intersections of intersection k.
$L_{i,l}^t$	The vehicle queue length at lane l and time step t for agent i.
$N_{i,j}^{[t-T,t]}$	The amount of the green waves between intersection i and j from time $t-T$ to t.
Δt_{action}	The minimum time interval of changing traffic phases.
α	$\alpha \in [0, 1]$ is the weight to balance the immediate reward and the reward shaping.
$\beta_{\text{inner}}, \beta_{\text{outer}}$	β_{inner} is the inner-loop step size; β_{outer} is the outer-loop step size.
β_{f}	β_{f} is the fine-tuning step size.
$\mathcal{L}, D_e$	$\mathcal{L}$ is the loss function; D_e is the memory replay.
R_{adapt}	The number of environments to do fine-tuning.

B Related Works

Coordination Strategies in TSC. To solve TSC problem, many multi-agent RL methods propose their coordination strategies to enhance traffic efficiency [4, 10, 11, 15].

Some methods use graph neural networks (GNN) to learn the interactions among agents. [3] uses graph convolutional networks (GCN) to automatically extract the traffic features of multiple intersections. Similar to [3], [12] uses graph attention networks (GAT) to coordinate multiple intersections by learning their dynamics of traffic conditions. Based on previous works, [9] proposes a multi-agent RL framework named STMARL. STMARL constructs spatio-temporal dependency among multiple traffic lights based on the adjacency graphs of roadnets. Moreover, [2] also proposes a spatio-temporal feature fusion method named MADRL-STFF. Compared with STMARL, MADRL-STFF constructs connected subnetworks to learn interactive relations among intersections in the subnetworks. These methods use GNN to achieve network communication to coordinate neighbor traffic signals.

For other methods, [13] proposes a method named PressLight to coordinate traffic signals in arterial networks. PressLight designs the reward function based on the theory of max pressure [6]. The pressure is defined as the number of vehicles on incoming roads minus the number of vehicles on outgoing roads. Therefore, PressLight can coordinate traffic signals by correlating incoming and outgoing traffic flows. [7] proposes a cooperative group-based multi-agent reinforcement learning framework. The framework introduces a novel action space including the pheromone-based regional green-wave control mode. When the vehicle queue length is greater than a threshold value, the regional green-wave control mode will coordinate two related traffic signals to generate green wave to alleviate traffic congestions. [5] proposes a decentralized-to-centralized framework. This framework divides a roadnet into multiple local regions. Each region is controlled by an RL agent. These local agents are coordinated by a global agent to achieve the centralized control. [18] proposes a Q-learning based method named QCOMBO. QCOMBO designs a novel objective function including independent learning and centralized learning. The centralized learning regards the global Q-value as a weighted sum of local Q-values based on max-plus algorithm. Besides, QCOMBO designs a regularizer to keep consistency between independent learning and centralized learning. [8] proposes a cooperative double Q-learning method named Co-DQL. Co-DQL uses mean-field approximation to learn the interactions among agents. Therefore, Co-DQL achieves local information sharing and a better coordinated strategy. These methods can well coordinate traffic signals from different perspectives. However, their coordination strategies can only adapt to their own traffic environments. They can not adapt to more complex environments.

Generalization to Different Traffic Flow Environments. To address the shortcomings of the methods mentioned above, recently, some methods are proposed to enhance generalization ability to adapt to different traffic flow envi-

ronments. These methods can be divided into two types: meta-reinforcement learning methods and inductive learning methods.

For meta-reinforcement learning methods, [16] proposes a value-based method called MetaLight. MetaLight designs a novel framework by combining individual-level adaptation and global-level adaptation. Based on this framework, MetaLight pays attention to finding a generalized model for any type of intersections and phase settings. [17] proposes a meta-reinforcement learning method called GeneraLight. GeneraLight clusters similar traffic environments based on average travel time. The traffic environments of the same cluster can be utilized to train a global parameter initialization. The global parameter initialization can be used to test new traffic environments. Compared with MetaLight, GeneraLight pays attention to improving the generalization ability of TSC models to adapt to different traffic flow environments. [19] proposes a meta variationally intrinsic motivated reinforcement learning method (MetaVIM). MetaVIM designs a novel intrinsic reward to keep the received observation and reward consistent regardless of neighbor agents. Based on the intrinsic reward, MetaVIM trains a stable policy to better adapt to various traffic environments.

For inductive learning methods, [1] proposes an inductive graph reinforcement learning method (IG-RL). IG-RL uses GCN to learn the representations of traffic roadnets from the lane level and the vehicle level. Besides, IG-RL uses the parameter sharing mechanism to train on small random networks and test on different networks without additional training. [14] proposes an inductive heterogeneous graph multi-agent actor–critic algorithm (IHG-MA). IHG-MA uses GNN to learn the representations of heterogeneous feature nodes to generate the embedding of each traffic signal agent. Then, IHG-MA uses a decentralized cooperative multi-agent actor-critic framework to generate the actions and Q-values of agents. Based on this framework, IHG-MA can be smoothly transferred to new synthetic and real-world datasets.

The meta-reinforcement learning methods and inductive learning methods mentioned above enhance the generalization performance of TSC models from different perspectives. However, these methods can not properly decide when to coordinate traffic signals according to traffic conditions. Therefore, this paper proposes Meta-CSLight to combine independent learning and neighbor-aware learning to adapt to light and heavy traffic conditions, respectively. Besides, a novel reward shaping is designed to enhance traffic efficiency by encouraging adjacent intersections to generate more green waves.

C Datasets

– $Hangzhou_{4\times4}$: This dataset is collected by roadside surveillance cameras near 16 intersections in Gudang Sub-district, Hangzhou. Based on this dataset, 10 different traffic flow datasets can be generated by WGAN when Wasserstein distance (W-distance) is set to 0. W-distance denotes the difference between generated data distribution and real data distribution. It represents that the generated 10 datasets have the same data distribution with

$Hangzhou_{4\times4}$. The generated 10 datasets ($Hangzhou_{4\times4,dis=0}$) can be used as the train set in the meta-training step. After the meta-training step, to test the efficiency of the trained f_{θ_0}, 20 traffic flows in the same roadnet are generated by setting W-distance to 0.1 ($Hangzhou_{4\times4,dis=0.1}$) based on $Hangzhou_{4\times4}$.

- $Jinan_{3\times4}$: This dataset is collected in Dongfeng Sub-district, Jinan. Based on this dataset, 20 traffic flows ($Jinan_{3\times4,dis=0}$) are generated to test the efficiency of the meta-learner.
- $Atlanta_{1\times5}$: This dataset contains 5 intersections in Atlanta. Based on this dataset, 20 traffic flows ($Atlanta_{1\times5,dis=0}$) are generated as the test set.
- $Syn_{3\times3}$: This dataset is a 3×3 traffic roadnet. The traffic flow in this roadnet is sampled from a Gaussian distribution. Based on this dataset, 20 traffic flows ($Syn_{3\times3,dis=0}$) are generated as the test set.

D Parameter Settings

For the hyperparameter setting, the number of episodes is set to 100. The inner-loop step size $\beta_{\text{inner}} = 0.001$. The outer-loop step size $\beta_{\text{outer}} = 0.25$. The discount factor $\gamma = 0.8$. The number of environments R_{adapt} to do fine-tuning is set to 10. The fine-tuning step size $\beta_f = 0.001$. The fine-tuning epoch $Ft = 30$. The weight $\alpha = 0.2$ to balance the immediate reward and the reward shaping.

The experiments are evaluated on Ubuntu 16.04. GeForce GTX 1080Ti is the graphics card. Python 3.6 and Cityflow 0.1 are our main software environment.

E Effect of Independent Learning and Neighbor-Aware Learning

To compare the effectiveness of independent learning and neighbor-aware learning, the experiments are conducted by removing each component from Meta-CSLight. The experimental results are shown in Table E2 and Table E3 with and without fine-tuning, respectively. Compared with the two variants of Meta-CSLight: Meta-CSLight w/o NL and Meta-CSLight w/o IL, Meta-CSLight outperforms the two variants by 73.42% and 5.27% on average with fine-tuning, respectively. Without fine-tuning, Meta-CSLight outperforms the two variants by 65.10% and 3.42% on average, respectively. It indicates the effectiveness of independent learning and neighbor-aware learning in Meta-CSLight.

Besides, some observations can be found as follows:

- The experimental results demonstrate that the neighbor-aware learning model is more effective than the independent learning model. As shown in Table E2 and Table E3, in terms of average travel time, the increases of Meta-CSLight w/o NL are more than those of Meta-CSLight w/o IL on all the datasets. It indicates that the neighbor-aware learning model plays a more important role than the independent learning model in Meta-CSLight. The reason may be that the neighbor-aware learning model learns the traffic features from

heavy traffic intersections. The traffic signals in these intersections are difficult to be controlled. Therefore, when there is only an independent learning model to be trained in a traffic flow environment, it is hard for the model to converge because of the large difference of traffic flow distributions between light and heavy intersections. Nevertheless, the independent learning model is also essential for Meta-CSLight, because the neighbor-aware learning model can not merge the traffic features of all the intersections.

– Meta-CSLight w/o NL achieves better performance before fine-tuning. Without fine-funing, Meta-CSLight w/o NL outperforms the method with fine-tuning by 6.15% on average. The reason may be that Meta-CSLight w/o NL only uses an independent learning model to learn traffic flow environments. This model will be trained by both light and heavy traffic flows. It is likely to overfit one of these traffic flows. Therefore, in fine-tuning case, Meta-CSLight w/o NL is hard to adapt to various complex traffic environments because of overfitting.

Table E2. Ablation study of independent learning and neighbor-aware learning with fine-tuning w.r.t average travel time (the lower the better).

	Hangzhou	*Jinan*	*Atlanta*	*Syn*
Meta-CSLight w/o NL	1005.39	1502.52	404.52	1542.13
Meta-CSLight w/o IL	376.27	268.83	195.14	151.80
Meta-CSLight	**355.30**	**262.25**	**178.46**	**144.96**

Table E3. Ablation study of independent learning and neighbor-aware learning without fine-tuning w.r.t average travel time (the lower the better).

	Hangzhou	*Jinan*	*Atlanta*	*Syn*
Meta-CSLight w/o NL	941.58	1390.84	388.29	1437.15
Meta-CSLight w/o IL	384.18	291.70	279.64	170.29
Meta-CSLight	**363.80**	**289.64**	**267.83**	**164.44**

F Effect of Reward Shaping

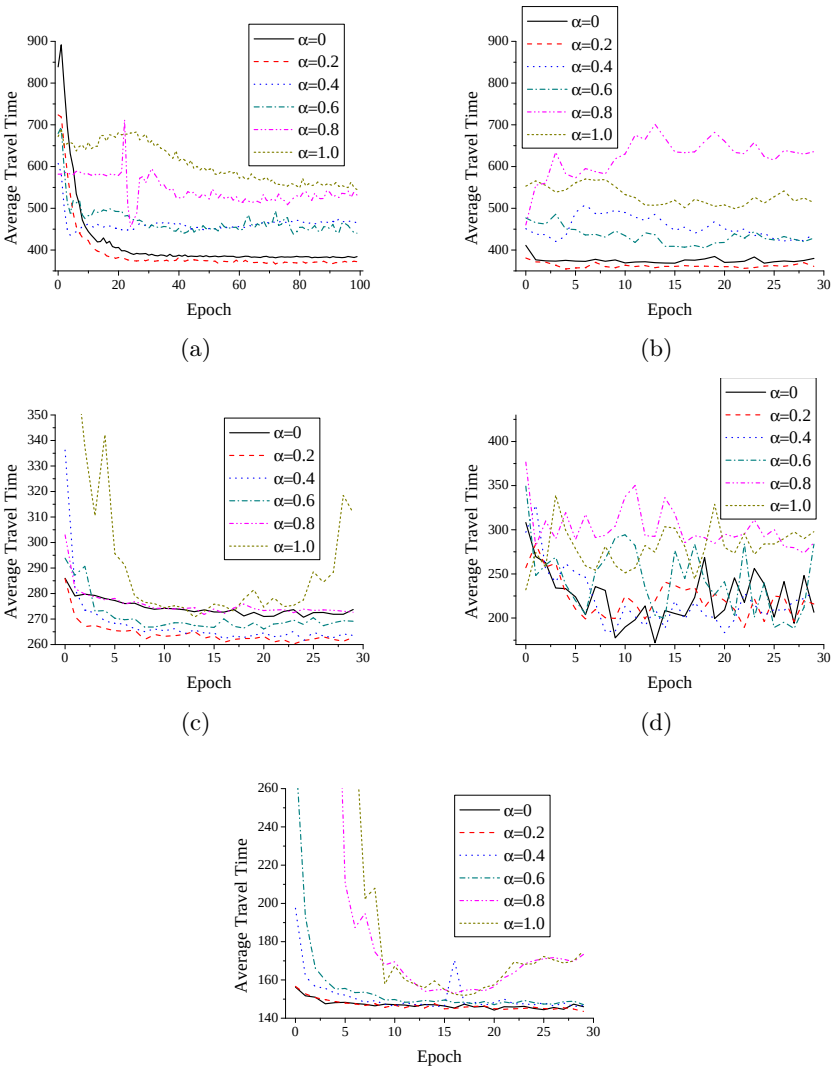

Fig. F1. Convergence curves by setting different weight α in the meta-training step and the fine-tuning stage. (a) The meta-training step on $Hangzhou_{4\times4,dis=0}$. (b) The fine-tuning stage on $Hangzhou_{4\times4,dis=0.1}$. (c) The fine-tuning stage on $Jinan_{3\times4,dis=0}$. (d) The fine-tuning stage on $Atlanta_{1\times5,dis=0}$. (e) The fine-tuning stage on $Syn_{3\times3,dis=0}$.

In Meta-CSLight, the reward shaping can enhance the traffic efficiency by encouraging more green waves. This paper uses the weight $\alpha \in [0, 1]$ to balance the immediate reward and the reward shaping. To investigate the sensitivity of α, in the meta-training step and the fine-tuning stage, the convergence curves are plotted by setting $\alpha = 0, 0.2, 0.4, 0.6, 0.8, 1$, respectively. The convergence curves are shown in Fig. F1.

The experimental results demonstrate that Meta-CSLight can converge to the best performance by setting $\alpha = 0.2$. In the meta-training step, in terms of average travel time, Meta-CSLight can converge to the best results in 20 episodes on $Hangzhou_{4\times4,dis=0}$ by setting $\alpha = 0.2$. In the fine-tuning stage, Meta-CSLight can converge to the best results in 5, 23 and 30 episodes on Hangzhou, Jinan and Synthetic datasets by setting $\alpha = 0.2$, respectively. On Atlanta dataset, Meta-CSLight converges to the best result in 13 episodes by setting $\alpha = 0$. However, the convergence curve in last serveral episodes fluctuates wildly. Compared with this curve, Meta-CSLight can stably converge to the similar results by setting $\alpha = 0.2$. It indicates that the reward shaping can improve the convergence of Meta-CSLight on these datasets.

Besides, when $\alpha \geqslant 0.4$, it is hard to converge to better results for Meta-CSLight on these datasets. Maybe the traffic signals of the neighborhood of each agent will make the TSC decision unbalanced by selecting a larger α.

G Case Study

To further show the effect of the reward shaping, this section compares the number of green waves by using the reward shaping with not using it on $Hangzhou_{4\times4,dis=0}$ in the meta-training step. The experimental results are shown in Fig. G2.

It can be observed that Meta-CSLight with reward shaping achieves the highest number of green waves in 20 episodes. There are no noticeable increments after 20 episodes. When the reward shaping is removed from Meta-CSLight, in terms of the number of green waves, the performance is similar with the former in the beginning but slowly increases after 20 episodes. It indicates that the reward shaping can accelerate the convergence speed in the meta-training step.

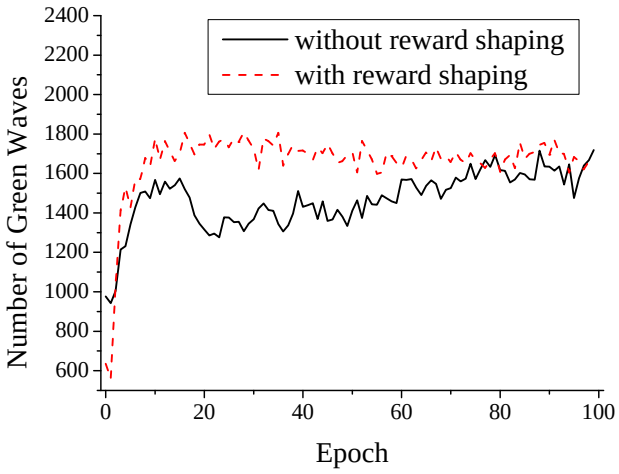

Fig. G2. The number of green waves with and without reward shaping on $Hangzhou_{4\times4,dis=0}$ in the meta-training step.

H Meta-Training Step of Meta-CSLight

Algorithm 1: Meta-Training Step of Meta-CSLight

Input: the traffic flow environments $\{V_e|e=1,2,...,M\}$, the inner-loop step size β_{inner}, the outer-loop step size β_{outer}, iteration epoch G, simulate time T_s

Output: the global parameter initialization θ_0

begin
  initialize the parameters of meta-learner f_{θ_0};
  for *episode* $= 1 : G$ **do**
    `/* Lines 4-12: Inner-loop stage */`
    for $e = 1 : M$ **do**
      $\theta_e \leftarrow \theta_0$;
      for $t = 1 : T_s$ **do**
        calculate the traffic pressures of intersections in V_e;
        generate training data for the independent learning model and the neighbor-aware learning model according to traffic pressures;
        generate and store the transitions into replay memory;
        update $\theta_e \leftarrow \theta_e - \beta_{\text{inner}} \nabla_{\theta_e} \mathcal{L}(f_{\theta_e}, D_e)$;
      end
    end
    `/* Line 13: Outer-loop stage */`
    $\theta_0 \leftarrow \theta_0 + \beta_{\text{outer}} \frac{1}{M} \sum_{e=1}^{M} (\theta_e - \theta_0)$;
  end
  return θ_0;
end

I Meta-Testing Step of Meta-CSLight

Algorithm 2: Meta-Testing Step of Meta-CSLight

Input: the global parameter initialization θ_0 from the meta-training step, the traffic flow environments $\{W_e|e = 1, 2, ..., R\}$, the number of environments R_{adapt} to do fine-tuning, fine-tuning step size β_f, fine-tuning epoch Ft, simulate time T_s

Output: average travel time of vehicles in test set

begin
  `/* Lines 2-12: Fine-Tuning */`
  for $e = 1 : R_{adapt}$ **do**
    $\theta_e \leftarrow \theta_0$;
    for $episode = 1 : Ft$ **do**
      for $t = 1 : T_s$ **do**
        calculate the traffic pressures of intersections in W_e;
        generate training data for the independent learning model and the neighbor-aware learning model according to traffic pressures;
        generate and store the transitions into replay memory;
        update $\theta_e \leftarrow \theta_e - \beta_f \nabla_{\theta_e} \mathcal{L}(f_{\theta_e}, D_e)$;
      end
    end
  end
  `/* Lines 13-21: Test */`
  initialize the list rl to store the test results;
  for $k = 1 : R - R_{adapt}$ **do**
    initialize the list ra to store the test results;
    for $e = 1 : R_{adapt}$ **do**
      calculate the average travel time of W_k by f_{θ_e};
      store the result in ra;
    end
    $rl[k] = avg(ra)$;
  end
  return rl;
end

References

1. Devailly, F.X., Larocque, D., Charlin, L.: IG-RL: Inductive graph reinforcement learning for massive-scale traffic signal control. IEEE Transactions on Intelligent Transportation Systems pp. 1–12 (2021, In Press)
2. Du, X., Wang, J., Chen, S., Liu, Z.: Multi-agent deep reinforcement learning with spatio-temporal feature fusion for traffic signal control. In: ECML-PKDD 2021. pp. 470–485 (2021)

3. Nishi, T., Otaki, K., Hayakawa, K., Yoshimura, T.: Traffic signal control based on reinforcement learning with graph convolutional neural nets. In: 2018 21st International Conference on Intelligent Transportation Systems. pp. 877–883 (2018)
4. Rasheed, F., Yau, K.L.A., Noor, R.M., Wu, C., Low, Y.C.: Deep reinforcement learning for traffic signal control: A review. IEEE Access **8**, 208016–208044 (2020)
5. Tan, T., et al.: Cooperative deep reinforcement learning for large-scale traffic grid signal control. IEEE Transactions on Cybernetics **50**(6), 2687–2700 (2020)
6. Varaiya, P.: The max-pressure controller for arbitrary networks of signalized intersections. In: Advances in Dynamic Network Modeling in Complex Transportation Systems, pp. 27–66 (2013)
7. Wang, T., Cao, J., Hussain, A.: Adaptive traffic signal control for large-scale scenario with cooperative group-based multi-agent reinforcement learning. Transportation Research Part C: Emerging Technologies **125**, 103046 (2021)
8. Wang, X., Ke, L., Qiao, Z., Chai, X.: Large-scale traffic signal control using a novel multiagent reinforcement learning. IEEE Transactions on Cybernetics **21**(3), 1086–1095 (2020)
9. Wang, Y., et al.: STMARL: A spatio-temporal multi-agent reinforcement learning approach for cooperative traffic light control. IEEE Transactions on Mobile Computing pp. 1–15 (2020, In press)
10. Wei, H., Zheng, G., Gayah, V., Li, Z.: A survey on traffic signal control methods. arXiv preprint arXiv:1904.08117 (2019)
11. Wei, H., Zheng, G., Gayah, V., Li, Z.: Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation. ACM SIGKDD Explorations Newsletter **22**(2), 12–18 (2020)
12. Wei, H., et al.: CoLight: Learning network-level cooperation for traffic signal control. In: CIKM 2019. pp. 1913–1922 (2019)
13. Wei, H., et al.: PressLight: Learning max pressure control to coordinate traffic signals in arterial network. In: KDD 2019. pp. 1290–1298 (2019)
14. Yang, S., Yang, B., Kang, Z., Deng, L.: IHG-MA: Inductive heterogeneous graph multi-agent reinforcement learning for multi-intersection traffic signal control. Neural Networks **139**, 265–277 (2021)
15. Yau, K.L.A., Qadir, J., Khoo, H.L., Ling, M.H., Komisarczuk, P.: A survey on reinforcement learning models and algorithms for traffic signal control. ACM Computing Surveys **50**(3) (2017)
16. Zang, X., Yao, H., Zheng, G., Xu, N., Xu, K., Li, Z.: MetaLight: Value-based meta-reinforcement learning for traffic signal control. In: AAAI 2020. vol. 34, pp. 1153–1160 (2020)
17. Zhang, H., Liu, C., Zhang, W., Zheng, G., Yu, Y.: GeneraLight: Improving environment generalization of traffic signal control via meta reinforcement learning. In: CIKM 2020. pp. 1783–1792 (2020)
18. Zhang, Z., Yang, J., Zha, H.: Integrating independent and centralized multi-agent reinforcement learning for traffic signal network optimization. In: Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems. p. 2083–2085 (2020)
19. Zhu, L., Peng, P., Lu, Z., Wang, X., Tian, Y.: Variationally and intrinsically motivated reinforcement learning for decentralized traffic signal control. arXiv preprint arXiv:2101.00746 (2021)