# Hierarchically and Cooperatively Learning Traffic Signal Control

Bingyu Xu, Yaowei Wang, Zhaozhi Wang, Huizhu Jia, Zongqing Lu

2023.04.10
발표자 : 이성진

# Contents

## Introduction

## Method

Hierarchy
Adaptive Weighting
Decentralized Training

## Related Work

## Experiments

Settings
Experimental Results

## Preliminaries

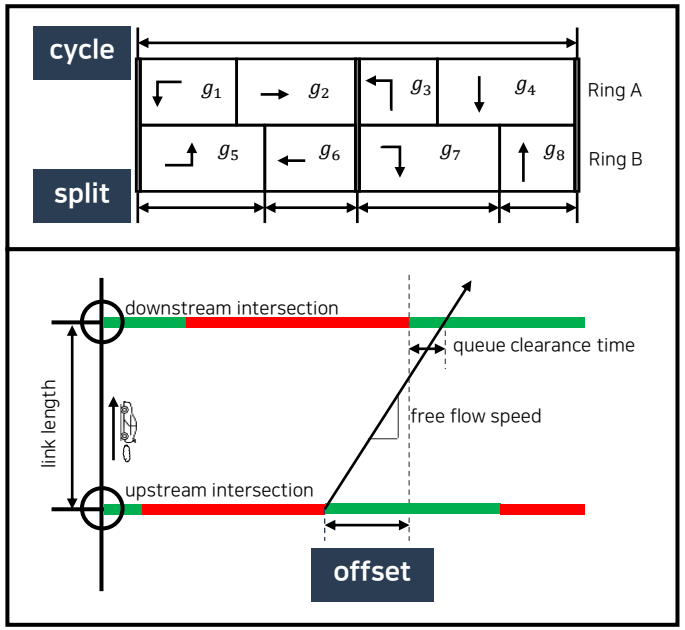
Intersection and Road Network
Movement and Phase
Average Travel Time

## Conclusion

# Conventional Traffic Signal Control

## Conventional pre-timed traffic signal control

- National Electrical Manufacturers Association (NEMA) Standards defines dual-ring based traffic signal control legacy with 3 signal control variable; cycle, split, offset.
- Use predetermined time interval (cycle), the duration of green signal for each phase (split), and the time difference between the start of green signals at adjacent intersections (offset) to regulate traffic flow.




[Cycle, Split, Offset Definition]

## Conventional real-time traffic signal control (Industrial View)

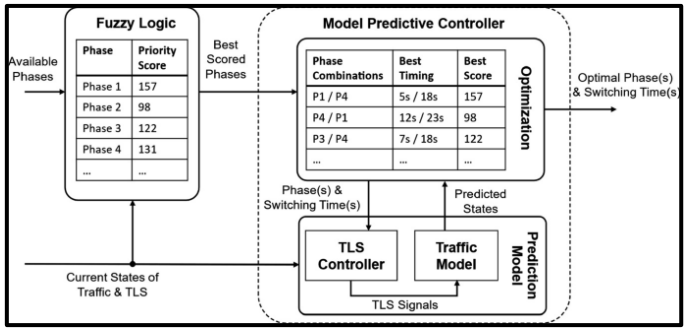
- Many trial to overcome the limitations of pre-timed control due to the high dynamics of road traffic in ITS domain industry; Adaptive Traffic Signal Control

TRL Software SCOOT TRL Software | scats DISTRIBUTION PARTNER | InSync | Synchro® 10 plus SimTraffic® Signal Timing & Analysis Software | swarco

- Most of them are based on separate optimizer for cycle, split, and offset which is heuristic algorithm, may result in a significant distance from the global optimum

## Conventional real-time traffic signal control (Scholar View)

- Optimization-based approaches using Model Predictive Control (MPC) predict real-time traffic condition and optimize the split of the next cycle.
- Fuzzy-logic based approaches design the rule of traffic signal control according to traffic demand of each approaches




[MPC based traffic signal control]

# Traffic Signal Control with Reinforcement Learning

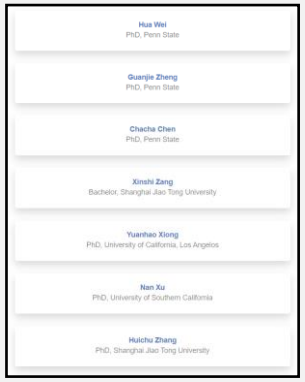


## Prof,Zhenhui Jessie Li research team

- Project Website: https://traffic-signal-control.github.io/

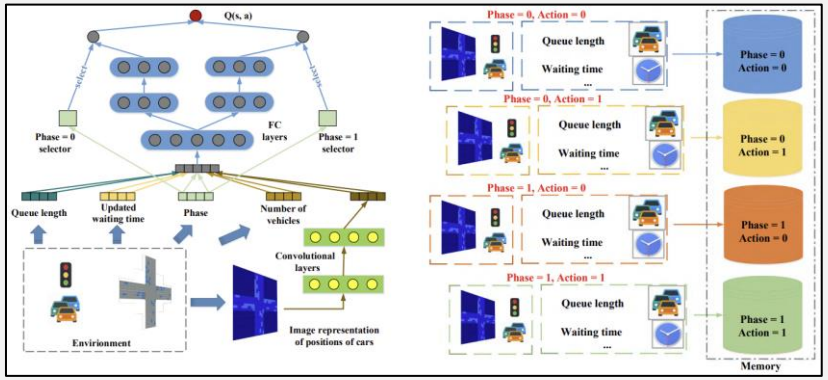
AAAI'20 Toward A Thousand Lights: Decentralized Deep Reinforcement Learning for Large-Scale Traffic Signal Control
AAAI'20 MetaLight: Value-based Meta-reinforcement Learning for Online Universal Traffic Signal Control
CIKM'19 Learning Phase Competition for Traffic Signal Control
CIKM'19 CoLight: Learning Network-level Cooperation for Traffic Signal Control
CIKM'19 Learning Traffic Signal Control from Demonstrations
KDD'19 PressLight: Learning Max Pressure Control to Coordinate Traffic Signals in Arterial Network
WWW'19 demo CityFlow: A Multi-Agent Reinforcement Learning Environment for Large Scale City Traffic Scenario
KDD'18 IntelliLight: A Reinforcement Learning Approach for Intelligent Traffic Light Control
Survey Survey on traffic signal control

Zhenhui (Jessie) Li
Penn State




## IntelliLight: 1st approach with RL

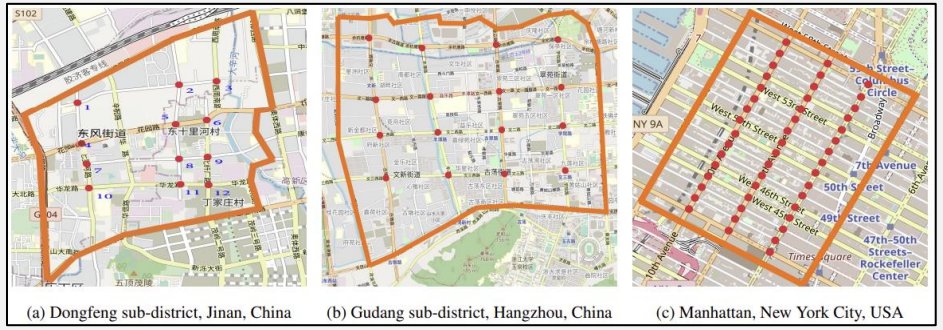
- 1st approach to control the traffic signal with reinforcement learning
- Simply DQN algorithm with simple action (change phase or not), simple phase (two phase), simple reward (delay, waiting time, queue)




[Model Architecture]

## Limitation of existing RL approaches

- Discrepancy between the target of RL method and the objective of traffic signal control.
- No considering how to cooperatively control traffic signals to optimize average travel time.




(a) Dongfeng sub-district, Jinan, China (b) Gudang sub-district, Hangzhou, China (c) Manhattan, New York City, USA

[Experimental Road Network]

# Related Work

## Previous Traffic Signal Control

### Conventional

- Roess et al. (2004)
  - ✓ Hand-crafted rules with fixed time
  - ✓ Fixed offset between intersections
- Mirchandani et al. (2001)
- Cools et al. (2013)
  - ✓ Pre-defined rules to trigger traffic signal by real-time traffic
- Varaiya (2013)
  - ✓ Coordination using max-pressure

### Simple RL based

- Simple Implementation
  - ✓ Wei et al. 2018; Zheng et al. 2019;Wei et al. 2019a, b; Chen et al. 2020
- RL engineering
  - ✓ State: Wei et al.2018; Xu et al. 2019
  - ✓ Action: Wei et al. 2018; Zheng et al. 2019; Chen et al. 2020
  - ✓ Reward: Zheng et al. 2019, El-tantawy et al. 2012; Wei et al. 2018
  - ✓ Algorithm: Aslani et al. 2017, Liang et al. 2018
  - ✓ Policy Adaptation: Xu el al. 2019; Zang et al. 2020

## How to consider interaction between intersections

- Centralized optimization
  - ✓ Pol et al. 2016
- Centralized learning, decentralized execution paradigms
  - ✓ QMIX (Rashid et al. 2018), QTRAN (Son et al. 2019)
- Learn cooperatively by acquiring information from adjacent intersection
  - ✓ El-tantawy et al. 2012; Nishi et al. 2018; Wei et al. 2019a,b; Chen et al. 2020

## Hierarchical RL

- Sub-goals
  - ✓ Vezhnevets et al. 2017; Nachum et al. 2018
- Options
  - ✓ Bacon et al. 2017; Frans et al. 2018
- Optimize multiple objectives
  - ✓ Jiang et al. 2019

# Preliminary

## General Setting for Traffic Signal Control

### Intersection and Road

- Each intersection is controlled by traffic signals
- East, South, West, North incoming approaches and East, South, West, North outgoing approaches
- Each incoming approach consists of three lanes
- From inner to outer, three lanes of each approach represent the “left-turn”, “straight”, “right-turn” directions

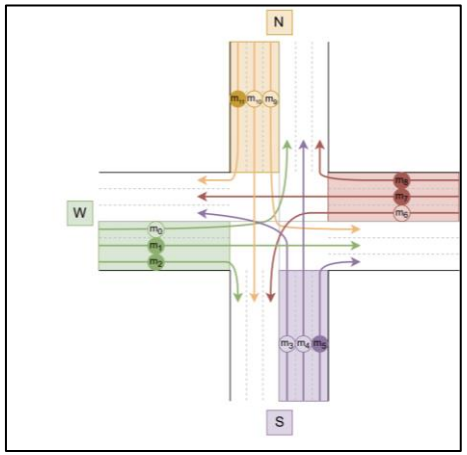
### Movements and Phase

- 12 movements of an intersection, $m_0$ to $m_{11}$
- 5 phases for a four-approach intersection
  - ✓ Phase 0: $m_2, m_5, m_8, and\ m_{11}$
  - ✓ Phase 1: $m_1, m_2, m_5, m_7, m_8, and\ m_{11}$
  - ✓ Phase 2: $m_1, m_2, m_5, m_7, m_8, and\ m_{11}$
  - ✓ Phase 3: $m_2, m_4, m_5, m_8, m_{10}, and\ m_{11}$
  - ✓ Phase 4: $m_2, m_3, m_5, m_8, m_9, and\ m_{11}$

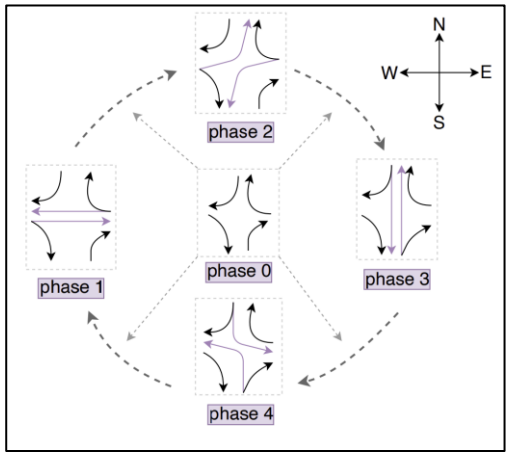
### Average Travel Time

- Most frequently used measure to evaluate the performance of traffic signal control
- 2 other version of average travel time: local travel time and neighborhood travel time
  - ✓ **local travel time** of an intersection is average time vehicles spent on the local area of the intersection
  - ✓ **neighborhood travel time** is average time vehicles spent on the neighborhood area of the intersection

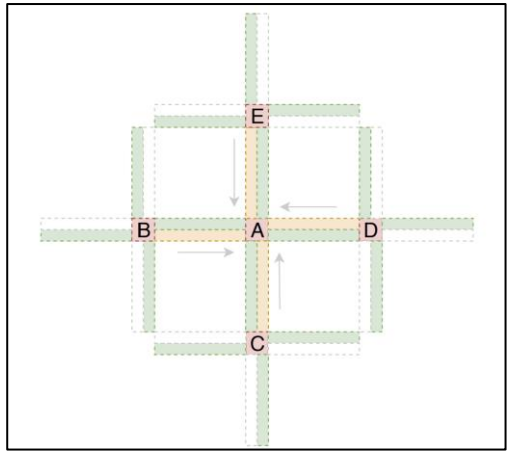
[Multi-intersection road network]



[Movement and Phase setting]



[Main loop consists of phases 1 to 4]



[Local; yellow, Neighbor; green]

# Hierarchy

## Hierarchy

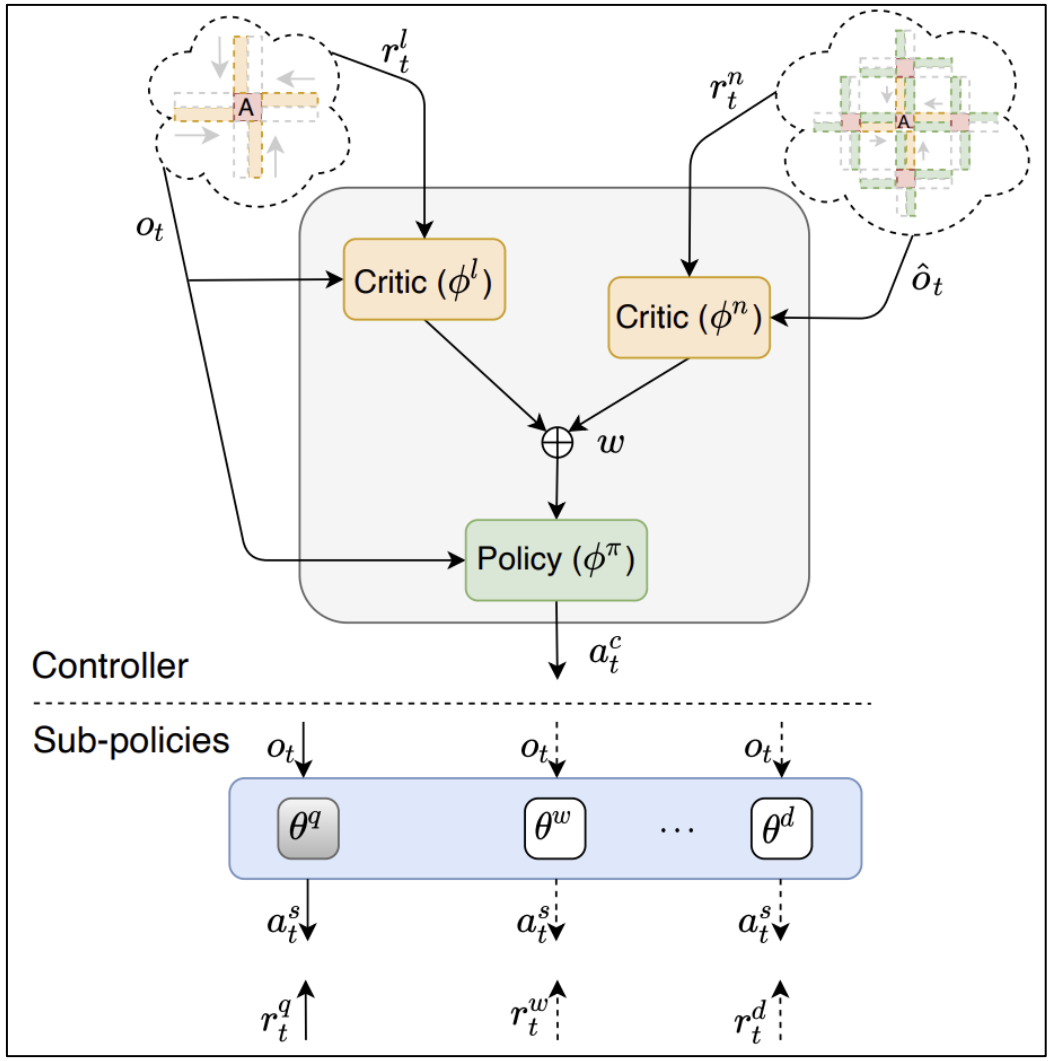
- Hierarchy consists of **a controller and several sub-policies**
- **Every T timesteps, the controller selects one of the sub-policies** and the chosen sub-policy directly act for the next T timesteps
- Controller focus on the long-term objective (**average travel time**) while the sub-policies concentrate on short-term targets.

### Specialized Sub-Policies

- Sub-policy of each agent gets an observation $o$, which is the concatenation of three vectors: current phase, next phase, and the number of vehicles of the incoming lanes.
- Rewards for the three sub-policies are the negatives of the sum of queue length ($r_q$), the sum of waiting time ($r_w$), and the sum of delay ($r_d$), respectively.

### Multi-Critic Controller

- Controller's reward is the negative of local travel time, but to optimize the average travel time in the network, an additional reward of neighborhood travel time need to be considered
- Adopt actor-critic RL method and multi-critic
  - ✓ Controller has two value network $V^l(o;\varphi^l)$ $and$ $V^n(\hat{o};\varphi^n)$
  - ✓ $\nabla\varphi^\pi = E\left[log\nabla_\pi\pi(a^c|o;\varphi^\pi)\left(\delta^l + w\delta^n\right)\right]$
    $where\ \delta^l = r^l + \gamma V^l(o';\varphi^l) - V^l(o;\varphi^l), \delta^l = r^l + \gamma V^l(o';\varphi^l)$




[HiLight Architecture]

# HiLight Training

## Adaptive Weighting

- Importance of optimizing neighborhood travel time might change under different traffic patterns.
- Adopt adaptive weighting mechanism (Lin et al. 2019) to enable the controller to learn the weight online to dynamically balance these two objectives during the learning process.
- $\nabla\varphi^{\pi} = E\left[log\nabla_{\pi}\pi(a^{c}|o;\varphi^{\pi})\left(\delta^{l} + w\delta^{n}\right)\right]$
- Let $L(\varphi_i^{\pi}) = L^{l}(\varphi_i^{\pi}) + wL^{n}(\varphi_i^{\pi}), where\ L^{l}(\varphi_i^{\pi}) = \delta^{l}, L^{n}(\varphi_i^{\pi}) = \delta^{n}$
- Aim to find the weight $w$ where $L^{l}$ decreased the fastest.
- $s_i(w)$ : the speed at which $L^{l}$ decreases at iteration $i$.
- $$s_i(w) = \frac{dL_i(\varphi_i^{\pi})}{di} \approx L^{l}(\varphi_{i+1}^{\pi}) - L^{l}(\varphi_i^{\pi}) = L^{l}\left(\varphi_i^{\pi} + \alpha\nabla_{\varphi_i^{\pi}}L(\varphi_i^{\pi})\right) - L^{l}(\varphi_i^{\pi})$$
$$\approx L^{l}(\varphi_i^{\pi}) + \alpha\nabla_{\varphi_i^{\pi}}L^{l}(\varphi_i^{\pi})^{T}\nabla_{\varphi_i^{\pi}}L(\varphi_i^{\pi}) - L^{l}(\varphi_i^{\pi})$$
$$= \alpha\nabla_{\varphi_i^{\pi}}\nabla_{\varphi_i^{\pi}}L^{l}(\varphi_i^{\pi})^{T}\nabla_{\varphi_i^{\pi}}L(\varphi_i^{\pi})$$
- $\boldsymbol{\nabla_w s_i(w) = \alpha\nabla_{\varphi_i^{\pi}}\nabla_{\varphi_i^{\pi}}L^{l}(\varphi_i^{\pi})^{T}\nabla_{\varphi_i^{\pi}}L(\varphi_i^{\pi})}$

## Decentralized Training

- Can be learned in a decentralized way since the observations of neighbors and neighborhood travel time can be get easily.
- Learned end-to-end with sub-polices (DQN) and the controller (PPO)

**Algorithm 1** HiLight training

Initialize controller $\phi$, sub-policies $\boldsymbol{\theta}$, and $w$ for each agent
**for** episode = $1, \ldots, \mathcal{M}$ **do**
  **for** agent =$1, \ldots, \mathcal{N}$ **do**
    The controller chooses one sub-policy $\theta$
    **for** $t = 1, \ldots,$ max-episode-length **do**
      The chosen sub-policy $\theta$ acts to the environment
      and gets the reward $\begin{cases} r_t^{q} & \text{if } \theta = \theta^{p}, \\ r_t^{w} & \text{if } \theta = \theta^{w}, \\ r_t^{d} & \text{if } \theta = \theta^{d}, \end{cases}$
      Update $\theta$ using update rule (1)
      **if** $t\%T = 0$ **then**
        The controller gets rewards $r_t^{l}$ and $r_t^{n}$
        Update $\phi^{l}$ and $\phi^{n}$
        Update $\phi^{\pi}$ with weight $w$ using (2)
        Update $w$ using (3)
        The controller re-selects one sub-policy
      **end if**
    **end for**
  **end for**
**end for**

[HiLight Pseudocode]

# Settings

## Datasets

- The four real datasets are from four cities including Manhattan, New York City in US and Jinan, Hangzhou, and Shenzhen in China.
- There are two sets of traffic data: one weekday and one weekend day, both of which have a time span of 10000 seconds

[Dongfeng sub-district, Jinan, China]


[Gudang sub-district, Hangzhou, China]

[Manhattan, New York City, USA]

[Fuhua sub-district, Shenzhen, China]

## Baselines

- Conventional methods
  - FixedTime (Koonce et al. 2008)
  - SOTL (Cools et al. 2013)
  - MaxPressure (Varaiya 2013)
  - PressLight (Wei et al. 2019a)
  - CoLight (Wei et al. 2019b)
- Ablation of HiLight
  - IQLQ: only queue length
  - IQLW: only waiting time
  - IQLD: only delay
  - LocalCritic: only local critic
  - NBHCritic: only neighborhood critic
  - StaticWeight: static weight, $w$

## Evaluation Metrics

- Average travel time of all vehicles
  - time span between entering and leaving the road network
- Throughput
  - number of vehicles which have finished their routes over the course of the simulation

# Experimental Results

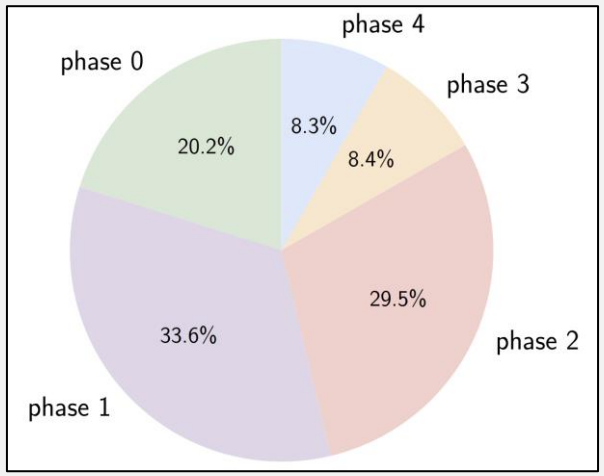
## Hyperparameters

| Hyperparameter | HiLight | PressLight | CoLight |
|---|---|---|---|
| discount ($\gamma$) | 0.9 | 0.8 | 0.8 |
| batch size | 128 | 20 | 20 |
| buffer capacity | 2048 | $1 \times 10^4$ | $1 \times 10^4$ |
| sample size | 512 | 1000 | 1000 |
| $\epsilon$ and decay | 0.4/0.97 | 0.8/0.95 | 0.8/0.95 |
| $T$ | 50 | – | |
| optimizer | Adam | RMSprop | RMSprop |
| learning rate | 0.0001 | 0.001 | 0.001 |
| learning rate of $V^l$ | 0.001 | – | |
| learning rate of $V^n$ | 0.001 | – | |
| # convolutional layers | | – | 3 |
| # MLP layers | 3 | 4 | 2 |
| # MLP units | | (32, 32) | |
| # MLP layers of $V^l$ and $V^n$ | 3 | – | |
| # MLP units $V^l$ and $V^n$ | (32, 32) | – | |
| MLP activation | | ReLU | |
| initializer | | random normal | |

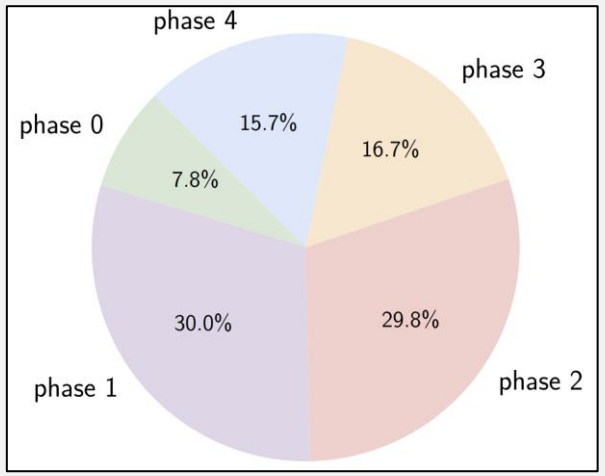
## Performance

| Method | Average Travel Time (seconds) | | | | | Throughput | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Jinan | Hangzhou | Manhattan | Shenzhen (D) | Shenzhen (E) | Jinan | Hangzhou | Manhattan | Shenzhen (D) | Shenzhen (E) |
| FixedTime | 749 | 762 | 1649 | 3821 | 3886 | 3399 | 1947 | 192 | 32 | 187 |
| SOTL | 1350 | 1413 | 1686 | 3854 | 3972 | 1480 | 2119 | 148 | 20 | 147 |
| MaxPressure | 397 | 352 | 443 | 3838 | 3873 | 5465 | 10428 | 2349 | 26 | 198 |
| PressLight | 382 | 425 | 1369 | 2220 | 1444 | 5517 | 9971 | 587 | 841 | 2246 |
| CoLight | 320 | 359 | 278 | 1574 | 1099 | 5688 | 10394 | 2615 | 1267 | 2999 |
| IQLQ | 381 | 386 | 292 | 531 | 665 | 5516 | 10251 | 2568 | 2126 | 3581 |
| IQLW | 437 | 470 | 866 | 817 | 758 | 5424 | 9673 | 1381 | 2197 | 3555 |
| IQLD | 452 | 736 | 526 | 797 | 795 | 5059 | 7279 | 1907 | 2140 | 3082 |
| LocalCritic | 346 | 343 | 271 | 395 | 440 | 5665 | 10529 | 2637 | 2370 | 3849 |
| NBHDCritic | 485 | 516 | 412 | 574 | 453 | 5137 | 9217 | 2228 | 2200 | 3826 |
| StaticWeight | 313 | 260 | 262 | 379 | 285 | 5725 | 11011 | 2631 | 2377 | 3968 |
| **HiLight** | **290** | **252** | **251** | **337** | **223** | **5769** | **11063** | **2635** | **2482** | **4088** |

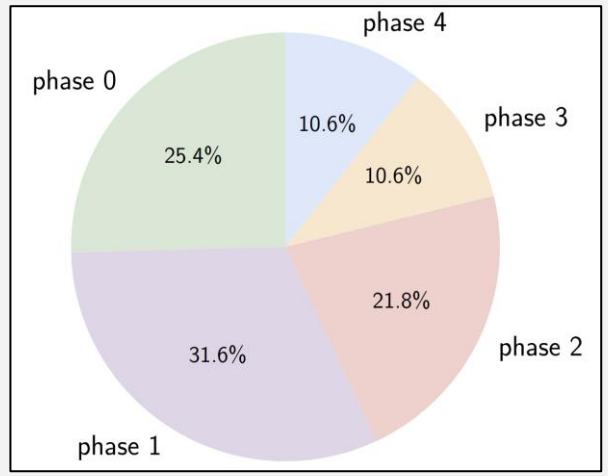
## Behaviors of the sub-policies




[phase ratio of $\theta^q$]



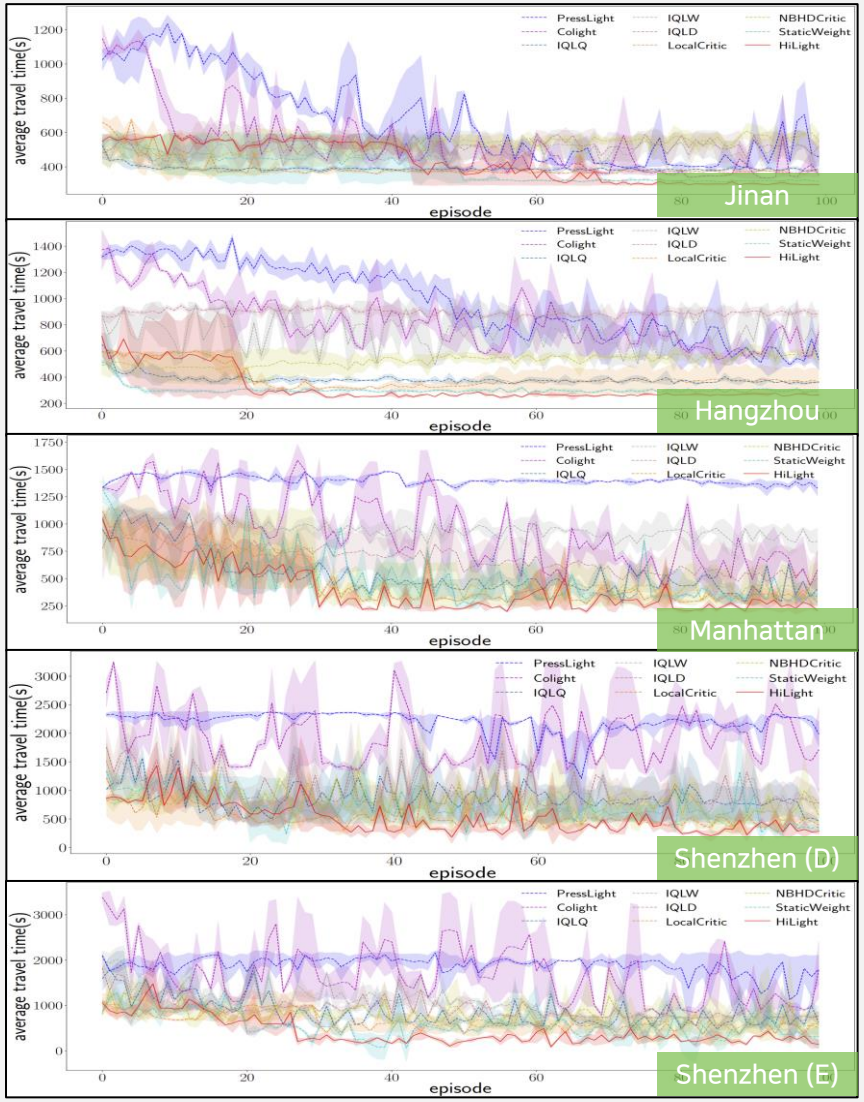

[phase ratio of $\theta^w$]



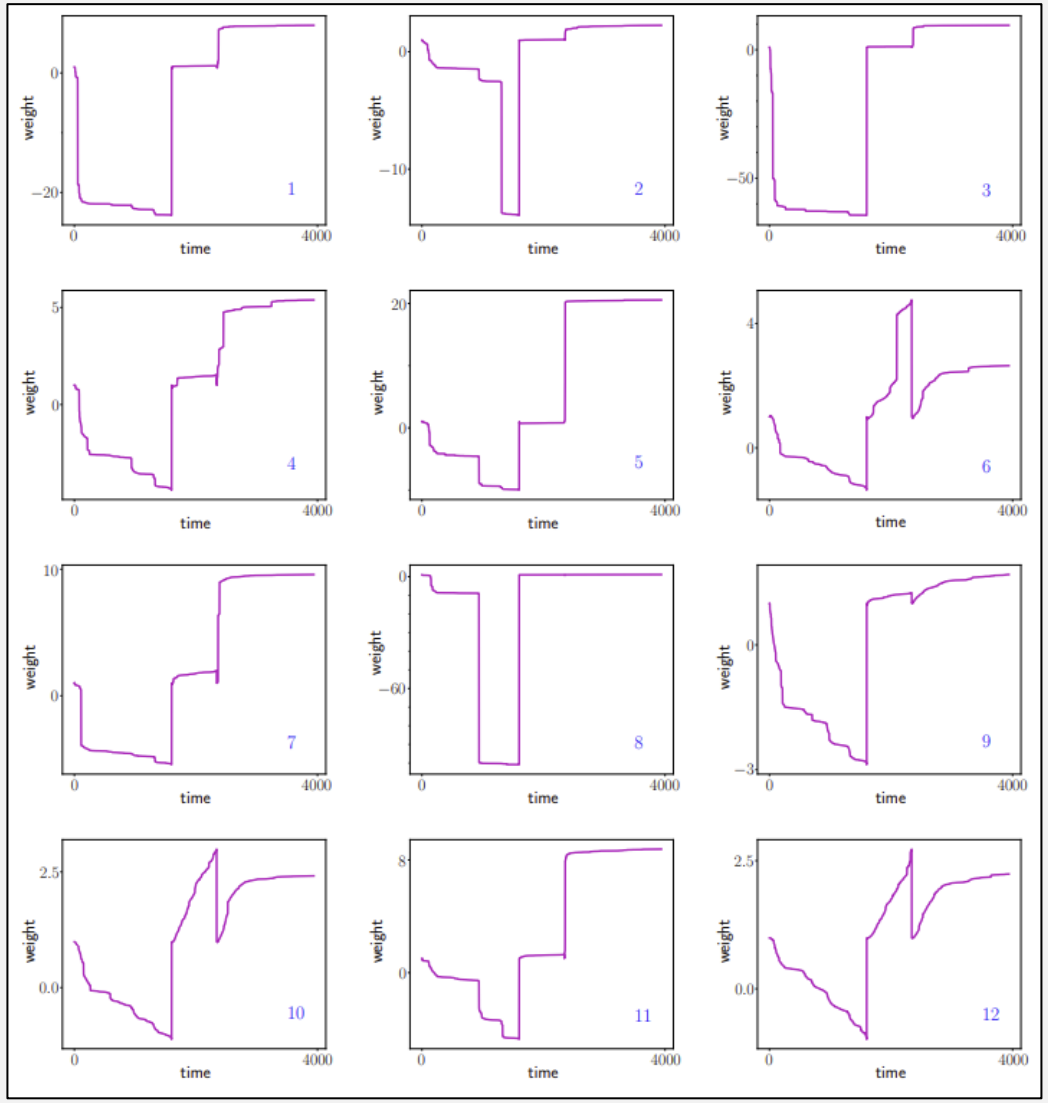

[phase ratio of $\theta^d$]

# Experimental Results

## Performance Comparison




## Weight ($w$) Change (at Jinan)






# Conclusion

## Conclusion

- We have proposed HiLight, a hierarchical RL method for cooperative traffic signal control.
- The controller minimizes local travel time and neighborhood travel time jointly with adaptive weighting by selecting among the sub-policies that optimize short-term targets.
- Cooperation among agents is encouraged by the optimization of neighboring travel time.
- It is empirically demonstrated in four real datasets that HiLight significantly outperforms the existing RL methods for cooperative traffic signal control.

## Acknowledgments

- This work is supported by NSFC under grant 61872009.

# 감사합니다