Real-time Reinforcement Learning in Traffic Signal System

Tianshu Chu

***Abstract*— Real-time optimization of a traffic signal system is a difficult decision-making problem with no considerable model given. To achieve this optimization, a perfect tradeoff should be made between the distributed learning in each individual intersection, and the integrated cooperation in the whole traffic grid. This paper explores such a tradeoff and designs an efficient traffic signal system by applying a hybrid ontology-based model. First, a model-based reinforcement-learning (RL) algorithm is induced. Second, a model of smart traffic signal system is designed with importing the RL algorithm into the background knowledge space of each signal-agent. Finally, a simulation of the model is conducted and the results are discussed.**

***Index Terms*—MAS, RL, smart infrastructure, ontology.**

I. Introduction

To mitigate the increasing traffic congestions during rapid urbanization, an optimal distribution of transportation resource is necessary. The real-time optimization of signal system is one fundamental approach. The research interests in this area have moved from modeling centralized control system to modeling hierarchically distributed control system due to more efficient and adaptive performance. To design high quality distributed signal system, two issues should be reviewed carefully:

- How to make each agent smarter to achieve the local optimum.
- How to make the whole multi-agent system (MAS) smarter to balance each local optimum and the global optimum.

Typical approaches to the first issue are RLs, such as Q-learning [1], TD-method [2], or model-based RL, which has more advantages for global optimum [3]. An RL example was given by B. Abdulhai *et al* [4]. Then an online learning can be implemented to achieve real-time optimization based on RL. For the second issue, usually evolutionarily stable strategy learning can be used in cooperative MAS [5]. However, this algorithm is practically difficult.

There are several examples considering both issues. S. Mikami and Y. Kakazu combined the individual RL with genetic searching algorithm to achieve global cooperation [6]. M. Choy, R. Cheu, and D. Srinivasan developed a more sophisticated model by implementing online learning with RL, learning rate and local weight adjustment, and evolutionary algorithm [7]. Later they added neural network as well [8]. However, to achieve the second issue, these models inevitably make the original RL much more complicated, leading to limited performances and low efficiencies. Actually, the simulations of these models were conducted only in small-scale traffic grid, and no real-time data was provided.

To design more efficient learning model, this paper uses an alternative approach to achieve the second issue: applying the ontology-based model [9], and introducing extra information sharing and communication processes in the dynamic ontology to make cooperative RL. First, an efficient model-based RL is developed using Q-function defined by M. Wiering [10]. Second, static and dynamic ontologies are designed considering the necessary information for RL and cooperation. Third, a Netlog simulation is conducted for large-scale traffic grid and different car-agents. Finally, the comparisons to other models of signal systems are given, and results are discussed.

II. Model-based Reinforcement Learning

A. Model objectives

The traffic grid is a square initialized by the road number (N) and block size (M). M is also the capacity of the queue before each intersection. Because each road has 2 directions, the total size will be a square of (2+M)*(N-1)+2. Each intersection $c \in C$ has four signals, and signal c_i controls traffic flow with dir_i, where dir = {N,W,S,E}. The decision space of each signal is U = {R (Red), G (Green)}. There are two basic traffic rules in this model which are shown in Fig.1:

- When the signal is red, the car can only turn right under yielding.
- When the signal is green, the car can go straight, turn right, or turn left under yielding

Three different car-agent types are designed to make the traffic system more realistic. Drunken car-agents do a random walk with legal actions and will arrive at the destination by change. Naïve car-agents are attempted to move closer to the destination. Smart car-agents are improved naïve car-agents, which can choose the optimal action to minimize expected waiting time in the next intersection.

B. Definition of Q- and V-function

The general goal of the whole signal system is minimizing the average cumulative waiting time of all car-agent on the way. So the Q-function can be defined as negative that value given their current state and the decision of each intersection. For example, if we have $u_t(c_1)$ = R, then we can simultaneously determine u_t = {R,G,R,G}. Also, each state can be identified

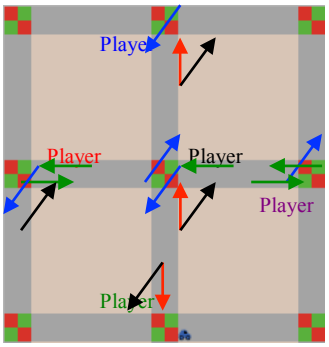

Fig.1 Traffic flows in the local system with decision profile (R,R,R,R). Black arrow shows the right-turn flow, green arrow shows the straight flow, blue arrow shows the left-turn flow, and red arrow shows the stopped cars. Note the incoming flow after the red flow also accounts for the congestion possibility.

with the current signal, car's position in the queue, and its destination. Then the Q- and V-functions are given as

$$Q_t(s_t, u_t), \text{ where } s_t = [c_t \; i \; q_t] \in S, \; s_{T+1} = \text{dest}, \; q_t = [1, M]. \quad (1)$$

$$V_{t+1}(s) = \max_u Q_t(s, u). \quad (2)$$

Both Q- and V-values are stored at each car-agent.

C. Definition of transition and reward functions

We can approximately define the transition function $P_t(s, u_t, v)$, $v \in S$, by treating all car-agents as drunken cars. Specifically, the transition functions for $v \neq s$ are listed below:

- when $u_t = R$, $q_t > 1$, q_t-1 is occupied: $P_t([c_t \; i \; q_t], u_t, [c_t \; i \; q_t\text{-}1]) = 1/2$.
- when $u_t = R$, $q_t > 1$, q_t-1 is empty: $P_t([c_t \; i \; q_t], u_t, [c_t \; i \; q_t\text{-}1]) = 1$.
- when $u_t = R$, $q_t = 1$: $P_t([c_t \; i \; q_t], u_t, [f(c_t, r(i)) \; r(i) \; M]) = 1/2$.
- when $u_t = G$, $q_t > 1$: $P_t([c_t \; i \; q_t], u_t, [c_t \; i\text{-}1 \; q_t]) = 1$.
- when $u_t = G$, $q_t = 1$: $P_t([c_t \; i \; q_t], u_t, [f(c_t, r(i)) \; r(i) \; M]) = P_t([c_t \; i \; q_t], u_t, [f(c_t, l(i)) \; l(i) \; M]) = P_t([c_t \; i \; q_t], u_t, [f(c_t, i) \; i \; M]) = 1/3$.

Where, r(i) (l(i)) is used to find next direction by turning right (left) with r(i) = i-1+4{i=1}, and $f(c_t, r(i))$ is used to find the next intersection. Note $P_t(\text{dest}, u, \text{dest})$ is always 1. The reward function is given as

$$R_t(s, u_t) = -\mathbf{E}1\{v = s\} = -P_t(s, u_t, s) \; . \quad (3)$$

D. Definition of updating rules

The original updating rule for Q-value is

$$Q_{t+1}(s_t, u_t) \leftarrow (1-\alpha) \; Q_t(s_t, u_t) + \alpha(R_t(s_t, u_t) + \beta V_t(s_{t+1})). \quad (4)$$

To simplify the problem, making $\alpha = 1$. Then in this model, the V-value can be alternatively expressed as

$$V_t(s) = \max_u \mathbf{E} \; (\Sigma_{\text{tau}=[t,T]} \; \beta^{\text{tau-t}} R_{\text{tau}}(s_{\text{tau}}, u_{\text{tau}}) + \beta^{T+1-t} \; R_{T+1}(s_{T+1})), \quad (5)$$

where $R_{T+1}(s_{T+1}) = 0$ if $s_{T+1} = \text{dest}$; otherwise $R_{T+1}(s_{T+1}) = -\infty$. So we can update the optimal decision per intersection with value iteration:

$V_{T+1}(c, i, q) \leftarrow R_{T+1}(c, i, q)$
for $t = T \rightarrow 0$ *until convergence*
$u_t^*(c, :, :) \in \text{argmax}_u \Sigma_{i,q}(R_t(s, u) + \beta \Sigma_v P_t(s, u, v) * V_{t+1}(v))$
$V_t(s) \leftarrow R_t(s, u_t^*(s)) + \beta \Sigma_v P_t(s, u_t^*(s), v) * V_{t+1}(v)$

In an agent-based model, this process can be approximately done with estimating the expected Q-value all the way before destination with $\beta = 0.95$. To check the performance, experiments are conducted in a small-scale traffic grid with N = 3, M = 3, and smart car number = 10. The estimated cumulative waiting time (absolute Q-value) of one car and corresponding average waiting time are shown in Fig.2, and Fig.3. Note all destinations can only be located on the road, and a car is also regarded as reaching destination if it reaches the neighbor patch on the road with opposite direction.

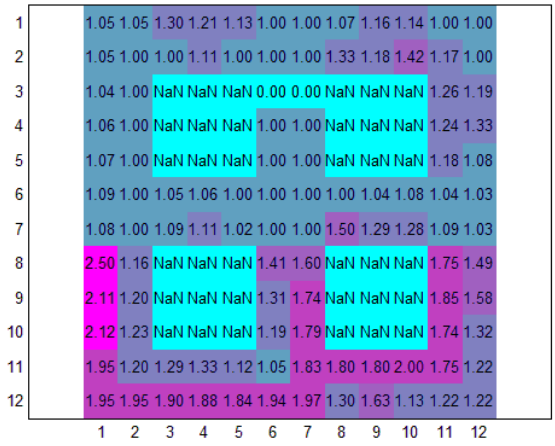

Fig.2 Q-value estimation in small-scale traffic grid

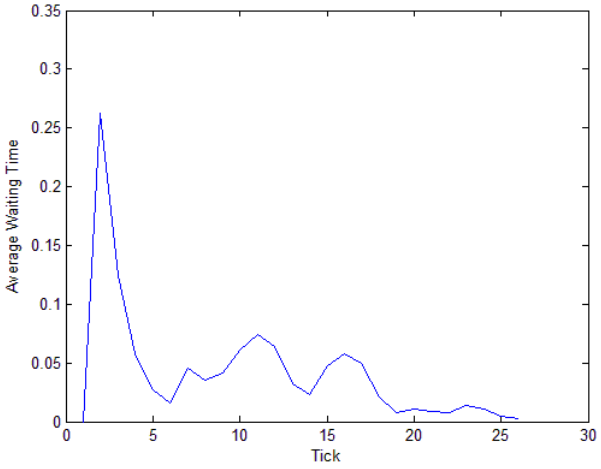

Fig.3 Average waiting time in small-scale traffic grid

E. Implementation of cooperative features

Since RL is time consuming and data reliable, it is only applied to coordinator-agent. During the RL process of coordinator-agent, intersection-agent can make naïve local-optimal decision to reduce its queue size. Also, coordinator-agent will send information to make balance between each child-agent. Therefore the whole decision-making process is guaranteed to be real-time cooperative.

III. Design of the Static Ontology

The similar static ontology is used here, which is shown in

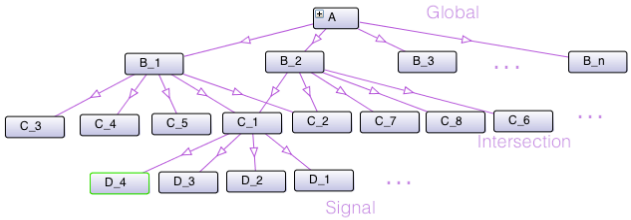

Fig.4 Static ontology

Fig.4. The only difference is that RL is used instead of greedy learning in the background knowledge space of coordinator-agents. Specifically, the goals for agents at each layer are listed in TABLE.1 below:

TABLE I. GOALS FOR AGENT LAYERS IN STATIC ONTOLOGY

Agent Layer	Goals
Signal	Conduct the decision from intersection-agent
Intersection	Greedily minimize the total queue size of child signal-agents; Consider the decision from coordinator-agent
Coordinator	Maximize the estimated total V-values of all car-agents inside this region (5 neighboring intersection-agents); Balance performances of child intersection-agents
Global	Present average and cumulative waiting time of car-agents inside this traffic grid for human-involved decision

Coordinator-agents paly main roles in this smart signal system: they seek for global optimal decisions to minimize the waiting time of cars on the way; they perform real-time communication with child-agents to make integrated local optimum. On the contrast, intersection-agents make simple and quick decisions to guarantee the adaption of the system. The global-agent in this model does not take responsibility for further global optimum because computing all car-based V-values in a combined state space of a large traffic grid is time-consuming. Instead, it just collects global data for result estimation.

The models in the background knowledge are listed in TABLE.2. Fictitious learning and Markov process are used in both intersection-agents and coordinator-agents. However, different decision-making models are applied for these two agents: intersection-agents perform greedy algorithm to minimize the estimated queue size at the next moment; coordinator-agents perform reinforcement learning to minimize the estimated total waiting time of all cars in the region. Due to different computing complication, intersection-agents can make real-time decision, while coordinator-agents can only update their decision within a period (2 seconds in this model). Furthermore, these two kinds of agents communicate with each other so intersection can also achieve balanced performance with its neighbors.

TABLE II. BACKGROUND KNOWLEDGE IN STATIC ONTOLOGY

Model	Description
Fictitious learning (FL)	Estimate the decision of intersection-agents
Markov process (MP)	Estimate the queue size
Greedy	Make the decision to minimize the estimated queue size in the region
Reinforcement learning	Make the decision to maximize the estimated V-values of all cars in the region

In short, intersection-agents make decision every second, based on the estimated queue size of itself and its neighbors (which is the partial information from coordinator-agents); coordinator-agents make decision every two seconds, based on the estimated cumulative waiting time of car-agents inside the region. Details of these two decision-making process are described below.

A. Decision-making process for intersection-agent

If we define c := intersection, i := signal, q := queue size, •^ := estimated value, H := relevant historical data, then the decision-making process can be described as:

For each $c \in \{Intersection\text{-}agents\}$
  For each $u \in \{G,R\}$
    For each $i \in \{Child\ signal\text{-}agents\}$
      $q_{t+1}\hat{}(c,i,u) \leftarrow MP(q_t(c,i),u)$
      $R_t(c,u) \leftarrow R_t(c,u) - q_{t+1}\hat{}(c,i)$
  For each $c_j \in \{Adjacent\ intersection\text{-}agents\}$
    $p_{t+1}\hat{}(c_j,u') \leftarrow FL(H(c_j))$
    $q_{t+1}\hat{}(c_j) \leftarrow \Sigma_u p_{t+1}\hat{}(c_j,u') * MP(q_t(c_j),u')$
  $R_t(c,G) \leftarrow R_t(c,G) - (0.83 * \Sigma_{1,3} q_{t+1}\hat{}(c_j) + 0.66 * \Sigma_{2,4} q_{t+1}\hat{}(c_j))$
  $R_t(c,R) \leftarrow R_t(c,R) - (0.83 * \Sigma_{2,4} q_{t+1}\hat{}(c_j) + 0.66 * \Sigma_{1,3} q_{t+1}\hat{}(c_j))$
  $u_t^*(c) \leftarrow argmax_u R_t(c,u)$

B. Decision-making process for coordinator-agent

To simplify this process, I assume all the cars are naïve-car agents so the transition space can be reduced significantly. Also, the expected waiting time at each intersection can be calculated using Geometric distribution. In fact, there are only two situations we need to consider (Fig.5)

- When the destination is on the right side ahead, car-agent can only choose the {turn right → …} path if the light is red; it can also choose the {go straight → …} path if the light is green.
- When the destination is ahead or on the left side ahead, car-agent always has longer waiting time if the light is red.

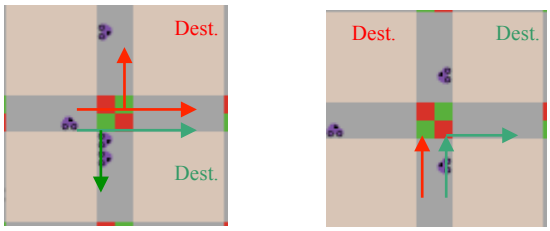

Fig.5 Action space of naïve cars before green light (left) and red light (right)

If we define Path := a set of possible paths presented as a sequence of intersections, V := negative cumulative waiting time to the destination, β = 1, then this process can be described as

For each $a \in \{Car\text{-}agents\ in\ the\ region\}$
  For each $c' \in \{Adjacent\ intersection\text{-}agents \cap Path\}$
    $u\hat{}(c') \leftarrow p_{t+1}\hat{}(c',u')$
    $V(a,c,u) \leftarrow \Sigma_{c'} P(a,c,u,c') * (-1/G\hat{}(c') + E[V(a,c',u\hat{}(c'))])$
$u^*(c) \leftarrow argmax_u \Sigma_a V(a,c,u)$

IV. Design of the Dynamic Ontology

The framework of the corresponding dynamic ontology is shown in Fig.6. Also, the carrying information is shown in TABLE.3.

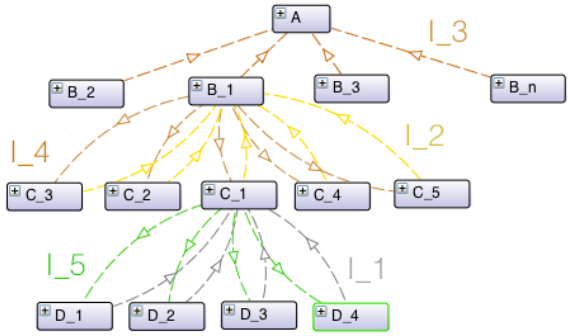

Fig.6 Dynamic ontology

TABLE III. Information Flows in Dynamic Ontology

Information	**Classifications**	
(Period [sec])	***Environmental info.***	***Experience info.***
I_1 (1)	q_c, $\delta_c(q_c+2)$ from c ∈ Children(C)	u^(c) from c ∈ Children(C) & dir(c) == 1
I_2 (1)	q_c $\delta_c(q_c+2)$ from c ∈ Children(B)	u^(c) from c ∈ Children(B)
I_3 (1)	q_c from c ∈ Children(A)	u* from c ∈ Children(A)
I_4 (2)	N/A	u* from c ∈ Parent(C) u^(c) from c ∈ Adjacent(C)
I_5 (1)	N/A	u* from c ∈ Parent(D)

V. Results and Analysis

In this section, the performance of this model will be estimated. First, a general comparison with regular signal system is conducted to confirm the efficiency. Second, the appropriate weight of local optimum is selected from plots. Third, it is compared with a naïve model built with greedy algorithm to ensure the difference between local optimum and global optimum in this signal system. Forth, the robustness in large-scale traffic grid is tested. Also, the overall performance with changing critical variables is visualized.

The default settings of the simulations are: run times = 100, N = 5, M = 5, car number = 60, car kind = naïve car, time horizon = 200 seconds, estimation standard = average waiting time of all cars. All the plots are actually the average values of 100 runs. Some settings may be changed later for the particular purpose of the experiment.

A. Comparison with regular signal system

To confirm the efficiency of this model for all car-agents, three car kinds are tested in this experiment. The performance of a regular signal system with fixed cycle time 20 seconds is provided for comparison. The results are shown in Fig.7 and Fig.8, for regular signal system and smart signal system designed with this model respectively. We can see in this model, the average waiting time is reduced significantly at the beginning, and drops fast when time elapses. The cumulative waiting time (area) also decreases because car-agents achieve their destinations in a short period.

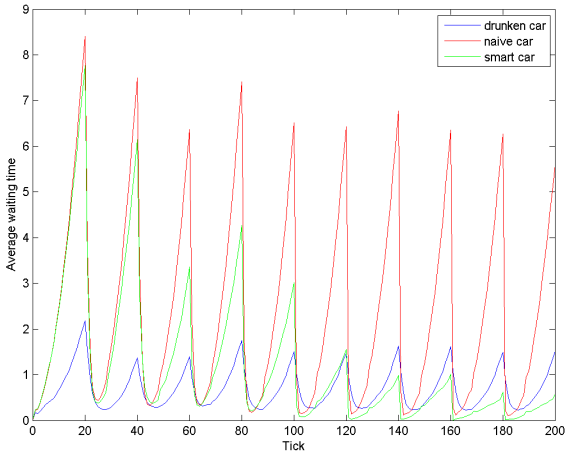

Fig.7 Average waiting time in a regular signal system

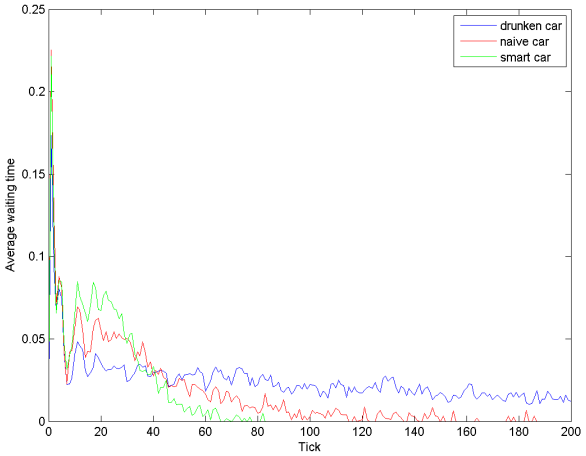

Fig.8 Average waiting time in a smart signal system designed with this model

B. Balance with local optimum

In this model, each intersection should balance its performance with its neighbors. However, how an appropriate tradeoff can be made between this cooperation and its local optimum? If we change the weight of local optimum part in the reward function of an intersection-agent, the result will be like Fig.9. Although heavy-weighted local optimum can achieve relatively low average waiting time at beginning, it extends cumulative waiting time of cars. Specifically, with weight = 1, all cars can achieve destinations after 80 seconds, while with weight = 2, the traveling time increases significantly to 170 seconds. So weight is chosen as 1 for this model.

C. Comparison with naïve local optimum

Although this model just considers neighboring optimum, in small-scale traffic grid given in default settings, it can be approximately regarded as global optimum. So we can compare

it with a naïve local optimum model, which only considers real-time minimizing of its queue length, to estimate how much a global optimum can improve the performance of this model. From the result shown in Fig.10, we can see when the traffic density is high, the performance of local optimum is very unstable and may occur irreversible congestion. Note the traffic density is calculated as initial car number / grid capacity, where the capacity is 4*M*N*(N-1).

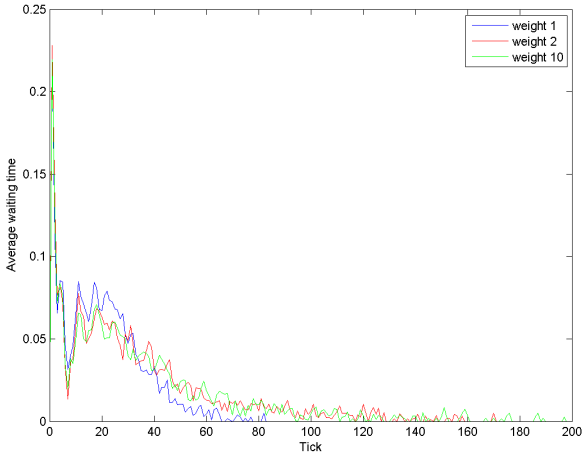

Fig.9 Performances with different weights of local optimum

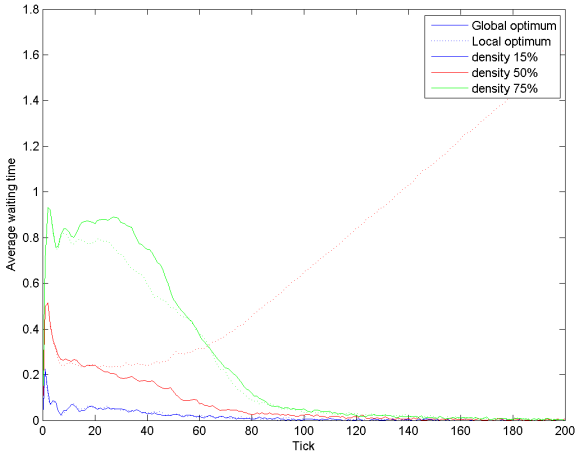

Fig.10 local and global optimums with different traffic densities

D. Performance in large-scale traffic grid

As the scale of traffic grid increases, this neighbor-cooperation becomes trivial, leading to a decrease global optimum. The performances of this model are also estimated in large-scale traffic grid with N = 10, 20 (Fig.11). When N = 10, the curve does not change very much; When N = 20, average waiting time increases at beginning, so cumulative waiting time increases significantly.

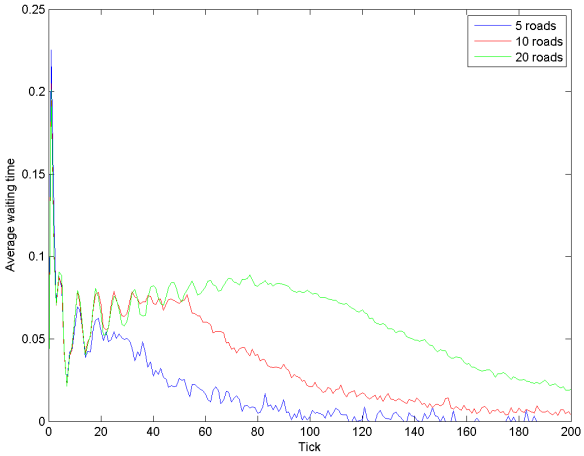

Fig.11 Performances in large-scale traffic grid

VI. SUMMARY

This paper designs a smart signal system by introducing a sophisticated reinforcement learning model and a neighbor-cooperation feature to the original ontology-based smart infrastructure design model. The efficient and stable performance of this model is demonstrated with a series of experiments. However, there is still improving space, such as combining cheap naïve local optimum with this model based on the current traffic density.

REFERENCES

[1] C.J.C.H. Watkins, “Learning from delayed rewards”, Doctoral dissertation, King’s College, Cambridge, 1989.

[2] R.S. Sutton, “Learning to predict by the methods of temporal differences”, Machine Learning, 3, 9-44, 1988.

[3] M.A. Wiering, “Explorations in efficient reinforcement learning”, Doctoral dissertation, University of Amsterdam, 1999.

[4] Baher Abdulhai, Rob Pringle, and Grigoris J. Karakoulas, "Reinforcement learning for true adaptive traffic signal control", Journal of Transportation Engineering 129.3, 278-285, 2003.

[5] Ana LC Bazzan, "A distributed approach for coordination of traffic signal agents", Autonomous Agents and Multi-Agent Systems 10.1, 131-164, 2005.

[6] Sadayoshi Mikami, and Yukinori Kakazu, "Genetic reinforcement learning for cooperative traffic signal control", IEEE World Congress on Computational Intelligence, Evolutionary Computation, 1994.

[7] Min Chee Choy, Dipti Srinivasan, and Ruey Long Cheu, "Cooperative, hybrid agent architecture for real-time traffic signal control", IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 33.5, 597-607, 2003.

[8] Srini Dipti Srinivasan, Min Chee Choy, and Ruey Long Cheu, "Neural networks for real-time traffic signal control", IEEE Transactions on Intelligent Transportation Systems, 7.3, 261-272, 2006.

[9] Tianshu Chu, Jie Wang, and James O. Leckie, “An Ontology-based Service Model For Smart Infrastructure Design”, Proceedings of IEEE Conference on Cloud and Service Computing, 2012.

[10] M. A. Wiering, "Multi-agent reinforcement learning for traffic light control", 1151-1158, 2000.