Reinforcement Learning based Urban Traffic Control System

Authors: Alvin Dey(MT18066) ,Pranay Raj Anand(MT18079), Swagatam Chakraborti(MT18146)

***Abstract*—This study demonstrates a reinforcement learning based traffic control system which can be applied in urban areas. The objective of the algorithms implemented is to calculate a phase time for traffic signal at an intersection so that the waiting time and average queue length at a stop-line of traffic signal is reduced. ANYLOGIC software is used for simulating a road network consisting of six intersections to test the efficiency of the algorithms for calculating an optimal phase time for each of the traffic signal. In the paper a comparative analysis of three reinforcement learning (RL) algorithm has been shown. It is observed that they fare better than static phase time implementation of traffic signals.**

I. Problem Statement

The unprecedented rise in the number of vehicles compared to expansion of road network have led to the problem of traffic congestion all across the world. "Traffic congestion during peak hours in four major cities — Delhi, Mumbai, Bengaluru and Kolkata — costs the economy Rs 1.47 lakh crore annually, according to a study conducted by global consultancy firm."-a TOI report April,2018. The obvious solution is to expand the road network but being a developing nation it is not that financially feasible. So the next best option is to implement a method which can regulate traffic properly and efficiently so as to decrease traffic congestion as much as possible. Currently, traffic signal phase times are static in India. Therefore, in this paper we try to implement RL algorithms to make the traffic signal phase times adaptive by using the traffic state information by sensors of traffic signal and the neighboring information given by traffic signals of other intersections. Using this adaptive traffic signal variation we aim to reduce the traffic congestion.

II. Literature Survey

Many research has been made in the field of reinforcement learning for traffic signal optimization problem. Different research have been made by proposing different learning type, different reward definition, state space definition, different action space, different simulator. Most of the previous research have been on the idea of varying the number of queued vehicles [3][6][7][8] and traffic flow [9][1] the most popular. There have been research work on varying the action spaces as the available signal phases [9][2] or only confined in varying the green phase [1][7][8]. The reward definition was mostly on the change in the average delay of the vehicles [4][6][7][8] and change in queued vehicles [1][7][8]. Below are the following observations that have been made. First, the majority of state definitions are abstractions of the traffic state which omit relevant information. These will not lead to the optimal state. Secondly the action space selection should be as much discrete as possible.

III. Model Specificaton & Data

Car: length(5m), max acceleration(1.6m/s^2), max deacceleration(4.4m/s^2),inflow rate(variable). *Intersection:* roads(4), lanes(1 forward and 1 backward), traffic signals(4).

IV. Methodology

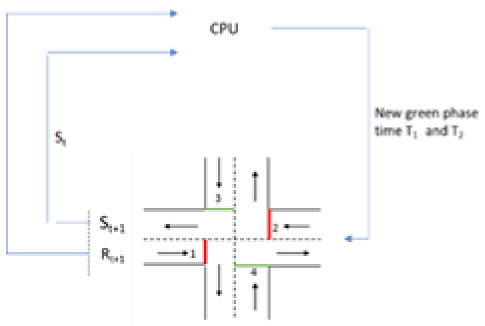


Figure 1 Reinforcement Learning Agent for Traffic Signal Control

In Figure 1, q_1 is sum of queue length at stop-line 1 and queue length at stop-line 2, q_2 is sum of queue length at stop-line 3 and queue length at stop-line 4, T_1 is the green phase time for stop-line 1 and stop-line 2, T_2 is the green phase time for stop-line 3 and stop-line 4

A. ***Rule Base***

1. *Calculation of green phase time*

$$T^{new}_{phase} = T^{old}_{phase} + \rho\Delta \qquad (2)$$

where ρ=i/12 where i=1,2,3….12

Δ is the factor that we consider while changing the phase time and it depicts the green phase utilization time i.e the percent of green phase time used by cars to cross the intersection.

$$\Delta = q - 0.2667 \times T^{old}_{phase} \quad (3)$$

$$\Delta^1_2 = Downstream_{current} - Downstream_{previous}$$

where Downstream is the average downstream from intersection 2

$$\Delta^2_2 = Upstream_{current} - Upstream_{previous}$$

where Upstream is the average Upstream from intersection 1

2. *Lane occupancy*

It is defined as the ratio of number of cars occupying the lanes to the number of cars that can occupy a lane.

$$L_{occupancy} = \frac{(q_1 + q_2)}{lane\ capacity} \quad (4)$$

.

3. *Traffic influx*

It defined as change in total cars in queue from previous state(5) to current state.

$$C_{influx} = (queue_{previous} - queue_{current})/queue_{previous} \quad (4)$$

where *queue*$=q_1+q_2$

B. Problem formulation for RL

1. *Traffic state formulation:* A traffic state is defined as the combination of lane occupancy(3) and traffic influx(4).On basis of lane occupancy: low($L_{occupancy}$<30%) , medium(30%<=$L_{occupancy}$ <70%), high($L_{occupancy}$>70%).On basis of traffic influx :decreasing(C_{influx}<0), stable(30%<=C_{influx}<60%), increasing(C_{influx}>=60%).So together they constitute 9 traffic states. (5)

2. *Action:* The ρ we mentioned in (2) can have 12 discrete values which will constitute our action space. We are using Epsilon-Greedy strategy for selecting an action. It selects an action considering the exploration and exploitation trade-off. (7)

3. *Reward:* The reward R given to agent for taking an action a on state s is defined below,

$$R(s,a) = (queue_{previous} - queue_{current}) \times 100 \qquad (6)$$

where queue is as defined in (4)

C. Algorithm

We are implementing following three RL algorithms:
1. Temporal Difference Q-Learning
2. SARSA
3. Monte-Carlo method

1. Temporal Difference Q-Learning

```
1. Set α=0.1, γ=0.8 and ϵ=0.6 for trade-off between convergence and precision.
2. For episodes=1 to 6
    2.1. Create a random initial traffic condition in simulator.
    2.2. On the basis of <$q_1,q_2,T_1,T_2$> given by simulator and the previous state traffic tuple <$q'_1,q'_2,T_1',T_2'$> the state s of current traffic is set using conditions specified in (5)
    2.3. Select an action a randomly or optimally on basis of Epsilon-Greedy strategy (7)
    2.4. Calculate the new phase times $T_1$ and $T_2$ using equation (2)
    2.5. Set a reward R(s,a) according to equation (6)
    2.6. Update the Q-value Q(s,a) in the Q-Matrix using the equation (1)
    2.7. Give the new phase times to the simulator.
    2.8. Set new traffic state as the current traffic state.
    2.9. Repeat from 2.2 until we have transited through 10 traffic states
```

2. SARSA

```
1. Set α=0.1, γ=0.8 and ϵ=0.6 for trade-off between convergence and precision.
2. For episodes=1 to 10
    2.1 Create a random initial traffic condition in simulator.
    2.2 On the basis of <$q_1^1,q_2^1,T_1^1,T_2^1$> and <$q_1^2,q_2^2,T_1^2,T_2^2$> given by simulator and the previous state traffic tuple values the state s of current traffic is set using conditions specified in (5)
    2.3 Select an action a randomly or optimally on basis of Epsilon-Greedy strategy (7)
    2.4 Calculate the new phase times $T_1$ and $T_2$ using equation (2)
    2.5 Set a reward R(s,a) according to equation (6)
    2.6 Based on the new phase times fed to the simulator a new state $s'$ is obtained.
    2.7 Select an action $a'$ randomly or optimally on basis of Epsilon-Greedy strategy (7)
    2.8 Update the Q-value Q(s,a) in the Q-Matrix using the equation (1)
    2.9 Set new action to be taken as $a'$.
    2.10 Set new traffic state as the current traffic state $s'$.
    2.11 Repeat from 2.2 until we have transited through 10 traffic states
```

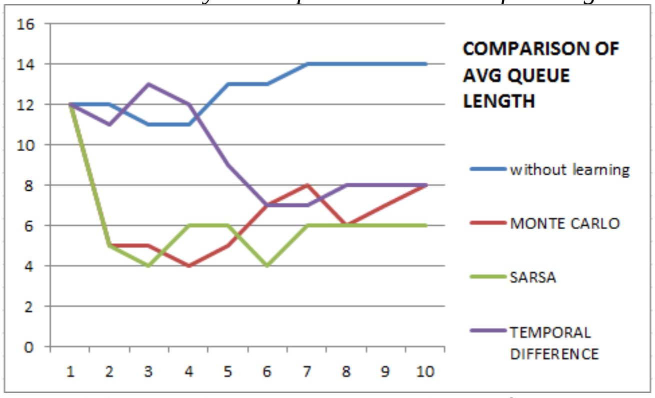
3. Monte-Carlo method

```
3.1 Initialize, for all s ∈ S where S is the state space , a ∈ A(s) where A belongs to action space of state s:
3.2 Q(s, a) ← arbitrary C(s, a) ← 0
3.3 π(s) ← a deterministic policy that is greedy with respect to Epsilon
3.4 Repeat forever: Generate an episode using any soft policy μ: $S_0, A_0, R_1, \ldots, S_{t-1}, A_{t-1}, R_t, S_t$ ($S_i$, $A_i$ are State and action at instant 't' and $R_i$ is the reward of $A_{i-1}$ on $S_{i-1}$)
3.5 G ← 0 W ← 1
3.6 For t = T − 1, T − 2, . . . downto 0:
    3.6.1 G ← γG + $R_{t+1}$ where γ is the standard discount factor used in RL algorithms.
    3.6.2 $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$ where C is the number of times $(S_t,A_t)$ appeared so far.
    3.6.3 $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + (W/C(S_t,A_t))*[G - Q(S_t, A_t)]$
    3.6.4 $\pi(S_t) \leftarrow \mathrm{argmax}_a\, Q(S_t, a)$ (Optimally)
```

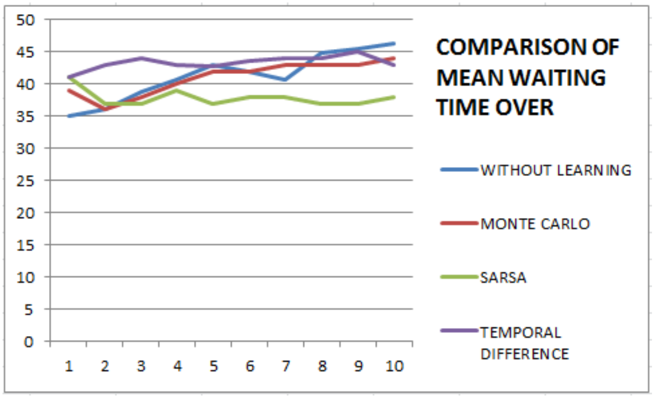
IV. RESULTS

After running our algorithms for 100 iterations(60 hours) each the following results were observed:

A. Peak hour analysis comparison on basis of waiting time



B. Peak hour analysis comparison on basis of waiting time



V. RESULT ANALYSIS

It is observed from the comparitive analysis between 3 RL algorithms and static implementation(i.e without any algorithm) that SARSA performs better than Temporal Difference Q-Learning by 41.33% and Monte Carlo method by 12.84%. All the RL algorithms outperformed the static timing implementation, SARSA bettering it by 56%. The average queue length was the distinguishing factor between all the algorithms but waiting time of vehicles failed to show any significant distinction between 3 RL algorithms.

VI. SUMMARY

The results show that the waiting time of vehicles at an intersection does not show much variation even using reinforcement learning. So it should not be used as a differentiating parameter to evaluate our algorithms because it encapsulates various intricate parameters like influx rate of vehicles etc.

VII. CONCLUSION

The proposed RL algorithms invloving the knowledge of neighbouring agents calculate traffic signal phase times in adaptive manner which performs better than staic timings at traffic signals. The average queue length and the average waiting time showed significant decrease by 56% and 26.66% respectively.

REFERENCES

[1] P. Balaji, X. German, and D. Srinivasan, "Urban traffic signal control using reinforcement learning agents," IET Intelligent Transport Systems, vol. 4, no. 3, pp. 177–188, 2010.

[2] S. El-Tantawy, B. Abdulhai, and H. 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, vol. 14, no. 3, pp. 1140–1150, 2013.

[3] T. L. Thorpe and C. W. Anderson, "Traffic light control using sarsa with three state representations," Citeseer, Tech. Rep., 1996.

[4] M. Wiering et al., "Multi-agent reinforcement learning for traffic light control," in ICML, 2000, pp. 1151–1158.

[5] E. Brockfeld, R. Barlovic, A. Schadschneider, and M. Schreckenberg, "Optimizing traffic lights in a cellular automaton model for city traffic," Physical Review E, vol. 64, no. 5, p. 056132, 2001.

[6] B. Abdulhai, R. Pringle, and G. J. Karakoulas, "Reinforcement learning for true adaptive traffic signal control," Journal of Transportation Engineering, vol. 129, no. 3, pp. 278–285, 2003.

[7] Y. K. Chin, N. Bolong, A. Kiring, S. S. Yang, and K. T. K. Teo, "Qlearning based traffic optimization in management of signal timing plan," International Journal of Simulation, Systems, Science & Technology, vol. 12, no. 3, pp. 29–35, 2011.

[8] M. Abdoos, N. Mozayani, and A. L. Bazzan, "Holonic multi-agent system for traffic signals control," Engineering Applications of Artificial Intelligence, vol. 26, no. 5, pp. 1575–1587, 2013.

[9] I. Arel, C. Liu, T. Urbanik, and A. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," IET Intelligent Transport Systems, vol. 4, no. 2, pp. 128–135, 2010.