Learning From Your Virtual Neighbors: a Reinforcement Learning Approach to Traffic Signal Control

Ana L. C . Bazzan[1],*, Henrique U. Gobbi[1]

[1]Computer Science, Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, RS, Brazil

Abstract

Many problems in traffic management and control are inherently distributed and/or require adaptation to the traffic situation. Hence, there is a close relationship to multiagent reinforcement learning. However, using reinforcement learning poses challenges when the state space is large. For instance, the learning task may take long. In order to accelerate this process, we allow agents that are similar to transfer experiences. We run experiments using a traffic network in which we vary the traffic situation along time. We compare our approach to the use of Q-learning without transfer of experiences, and show that there is an overall improvement in the number of stopped vehicles.

Keywords

Traffic Signal Control, Multiagent Reinforcement Learning

1. Introduction

Several approaches to adaptive traffic signal control exist, with reinforcement learning (RL) gaining in popularity. Here, learning agents are in charge of controlling the signals at a single intersection, in a distributed and decentralized way. One important aspect of such learning task is that traffic signal control is highly affected by actions of other agents. This is what makes multiagent RL much more challenging than single agent RL. Another challenge is the fact that the state space is very large. Hence, RL algorithms like Q-learning converge slowly. Various approaches to accelerate the learning task have been proposed, many of them are based on agents communicating with others to transfer their knowledge. However, they normally consider communication among neighbors only.

In the present paper we also explore the use of non-local information. By non-local we mean that agents that are not necessarily neighbors also exchange information. For this, we define and use a relationship graph, where junctions (intersections) of the traffic network that have similar pattern for a set of attributes are then connected. This means that non-local relationships (based on similarity) are formed and agents then exchange information about travel time and rewards, thus increasing the knowledge each agent has. This in time may help accelerate the learning process.

2. Background on traffic control via reinforcement learning

This section briefly presents underlying concepts on RL and on traffic signal control.

2.1. Reinforcement learning

In RL, an agent learns how to act in an environment interacting and receiving a feedback signal (reward) that measures how its action has affected the environment. The agent does not a priori know how its actions affect the environment, hence it has to learn this by trial and error (in an exploration phase). However, the agent should not only explore; in order to maximize the rewards of its action, it also has to exploit the gained knowledge. Thus, there must be an exploration-exploitation strategy that is

ATT'24: Workshop Agents in Traffic and Transportation, October 19, 2024, Santiago de Compostela, Spain

bazzan@inf.ufrgs.br (A. L. C. . Bazzan); hugobbi@inf.ufrgs.br (H. U. Gobbi)
http://www.inf.ufrgs.br/~bazzan (A. L. C. . Bazzan)
0000-0002-2803-9607 (A. L. C. . Bazzan)

to be followed by the agent. One of these strategies is ε-greedy, where an action is randomly chosen (exploration) with a probability ε, or, with probability 1-ε, the best known action is chosen, i.e., the one with the highest value so far (exploitation).

It is assumed that the agent has sensors to determine its current state and can then decide on an action. The reward is then used to update its policy, i.e., a mapping from states to actions. This policy can be generated or computed in several ways.

For the sake of the present discussion, we concentrate on a model-free, off-policy algorithm called Q-learning [1], which estimates so-called Q-values using a table to store the experienced values of performing a given action when in a given state. Hence Q-learning is a tabular method, where the state space and the action space need to be discretized.

In RL, the learning task is usually formulated as a Markov decision process (MDP), that defines the sets of states and actions, a transition function, and a reward function. Since the transition and the reward functions are unknown to the agent, its task is precisely to learn them, or at least a model for them.

In Q-learning, the value of a state s_t and action a_t at time t is updated based on Eq. 1, where $\alpha \in [0, 1]$ is the learning rate, $\gamma \in [0, 1]$ is the discount factor, s_{t+1} is the next state and r_t is the reward received when the agent moves from s_t to s_{t+1} after selecting action a_t in state s_t.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_a(Q(s_{t+1}, a)) - Q(s_t, a_t)) \tag{1}$$

When there are multiple agents interacting in a common environment, the RL task (thus, multiagent RL) is inherently more complex because agents' actions are highly coupled and agents are trying to adapt to others that are also learning. Despite this drawback, in many real-world problems, where the control is decentralized, it might not be possible to avoid a multiagent RL formulation. This is the case regarding the scenario we deal with in the present paper, namely control of traffic signals, whose basic concepts we briefly review next.

2.2. RL-based traffic signal control

Besides safety and other issues, one aim of a traffic signal controller is to decide on a split of green times among the various phases that were designed to deal with geometry and flow issues at an intersection. This can be done in several ways (for more details, please see a textbook such as [2]). In this paper, the controller is given a set of phases and has to decide which one will receive right of way (green light).

A phase is defined as a group of non-conflicting movements (e.g., flow in two opposite traffic directions) that can have a green light at the same time without conflict.

In its simplest form, the control is based on fixed times, whose split of the green time among the various phases can be computed based on historical data on traffic flow, if available. The problem with this approach is that it may have difficulties adapting to changes in the traffic demand. This may lead to an increase in the number of stopped vehicles. To mitigate this problem, it is possible to use an adaptive scheme and thus give priority to lanes with longer queues (or other measures of performance). Adaptive approaches based on RL were developed, as we discuss next.

3. Related work

Traffic control techniques stem mainly from the areas of control theory and operations research. More recently though, techniques from artificial intelligence and multiagent systems have also been employed, especially in connection with RL. RL is used by traffic signals to learn a policy that maps states (usually queues at intersections) to actions. Due to the number of works that employ RL for traffic control, the reader is referred to surveys [3, 4, 5, 6, 7]. In any case, those surveys show that there has already been a significant contribution of RL techniques to control of traffic signals. However, some issues arise, especially if tabular methods (e.g., the aforementioned Q-learning) are used. Tabular methods require agents to visit all state-action pairs. In order to accelerate this process, in the present paper we propose the use of transfer of experiencies among similar agents.

Other authors have addressed that issue in different ways. Next, we briefly review some of these approaches.

A first line of research does use tabular methods, with various levels of discretization of the state space, thus depicting different levels of performance of the learning task. In this class, well-known works include [8, 9, 10, 11], among others.

Other works avoid using tabular methods, substituting it for various kinds of function approximation techniques such as tile coding [12]; deep neural networks, such as DQN [13]; and linear function approximation [14].

Accelerating the RL task underlying traffic signal control was addressed using several techniques. For example, in [15], agents communicate using a hierarchy of supervising agents; in [16], k-nearest neighbors is employed to estimate the Q-values of an state by calculating the weighted average of the Q-value estimates of the k nearest states.

Finally, the idea of using a graph to establish the relationship among learning agents was used in traffic networks in [17, 18, 19]; however, these works aimed at driver agents rather than junctions/intersections.

4. Methodology

As mentioned, our approach is based on using (potentially non-local) communication to augment the information each traffic signal agent has. In the next sections, we detail the communication aspect (Section 4.3). For this purpose we employ a graph of relationships. Because this graph does not necessarily takes only agents that are neighbors into account, we call it a virtual graph (henceforth VG); this is explained in Section 4.1. The underlying MDP is explained in Section 4.2.

4.1. The Road and the Virtual Graphs

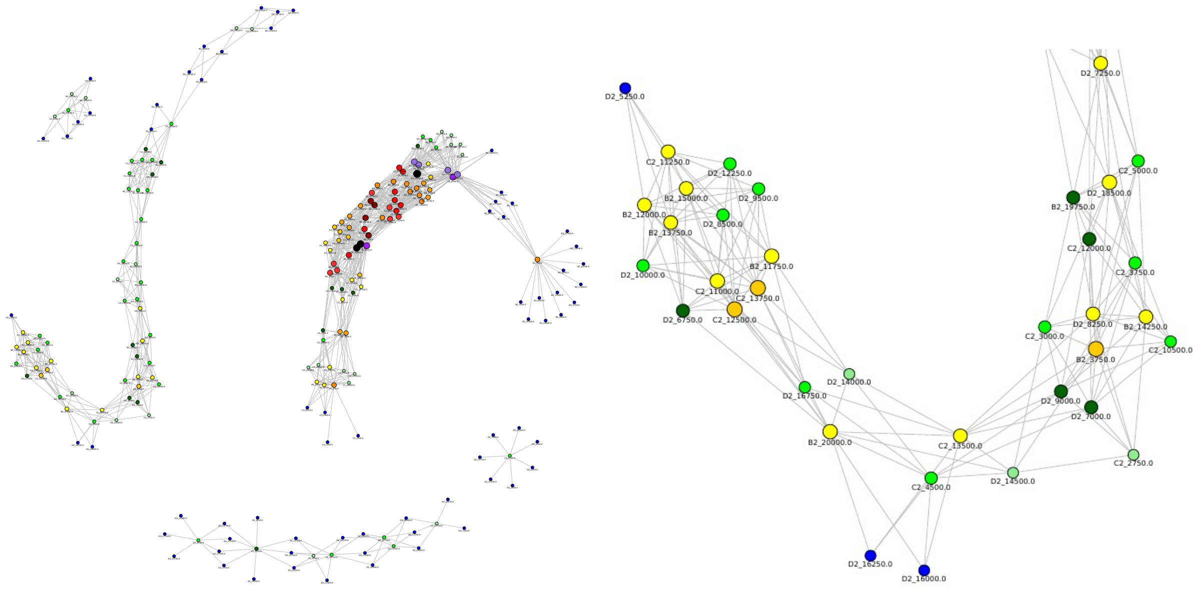
The road network, as conventionally used in traffic and transportation engineering, is a graph $G = (J, L)$, where J is the set of junctions (intersections), and L is the set of links. We use the term link, since it is more commonly used in traffic engineering (and then reserve the term edge for another graph, as described ahead). One instance of G is depicted in the next section (Fig. 2), where we discuss that scenario in more detail. Here, it suffices to note that intersections B2 and C2 (for instance) are geographical neighbors.

We also define another graph – the virtual graph –, which accounts for non-local information. In such graph, we connect two junctions $j_1 \in J$ and $j_2 \in J$, which are *not necessarily physically close* (as, e.g., D2 and B2 in Fig. 2), but that may have similar patterns in terms of the learning task. We call this a virtual graph denoted by $VG = (J, E)$, where J is as defined before, and E is the set of edges that connect two junctions that have similar patterns.

In order to define when two junctions are to be connected in VG, historical information is collected for a road network G. This information refers to several attributes: travel time over all links in an intersection, fuel consumption and several kinds of gas emissions such as CO, CO2, HC (hydrocarbon), PMx (particulate matter), and NOx. This information is collected per lane, per time interval, and then aggregated so that each junction has a single value associated with each of those attributes (in our case, we collect such data from a microscopic simulator). Moreover, such information is aggregated over a time window w_h. Then, values of all attributes are normalized between zero and one.

After the normalization, the values of the attributes for each two pairs of junctions are compared. If two junctions j_1 and j_2 have the same values for all attributes (given a tolerance value, i.e. $\pm\delta$), then an edge connecting j_1 and j_2 is inserted in the VG.

Fig. 1a shows an instance of such a virtual graph, whereas Fig. 1b depicts a zoom of that graph, where some relationships among similar junctions can be better seen. The labels of the vertices are formed by the junction ID plus the timestamp in which their values were found to be similar.



(a) Full VG.

(b) Zoom of a small part of the VG.

Figure 1: Instance of a virtual graph VG.

We stress that junction D2 experiences patterns of high traffic in both directions, which will only be experienced by junctions B2 and C2 later, when the traffic situation changes. Thus, it is possible to see that in Fig. 1b, junction D2 (at about time 5000) indeed has relationships with C2 and B2 at times around 10 to 13 thousand steps. This is just one example of such relationships. Overall, we can observe that, when a given junction has already experienced a given pattern, it is able to make a relationship with another one, thus allowing the passing along of past experiences.

4.2. MDP Formulation

As mentioned in Section 2, a RL learning task is formulated by an MDP. In our case, given a road network G, the set of states is defined over several dimensions. We note that this formulation is standard in the literature. Thus, at each time t that a signal controller agent has to make a decision, the agent observes a vector $s^t = [\rho, \tau, \Delta_1, ..., \Delta_{|L|}, q_1, ..., q_{[L]}]$, which describes the current state of the respective intersection. In this vector, $\rho \in \{0, 1\}$ is a variable that indicates the current active green phase; $\tau \in [0, 1]$ is the elapsed time of the current signal phase divided by the *maxGreenTime*; L is the set of all incoming links; the density $\Delta_l \in [0, 1]$ is defined as the number of vehicles in l divided by the total capacity of the link; $q_l \in [0, 1]$ is defined as the number of queued vehicles in l divided by the total capacity of the lane. As usual in the literature, a vehicle is considered to be queued if its speed is below 0.1 m/s.

Each agent chooses a discrete action a^t at each time step t in which it has to make a decision. For our scenario, all intersections have two phases. Thus, each agent has two actions: *keep* and *change*. The former keeps the green signal active, while the latter switches the current green light to another phase. The agents can only choose *keep* if the current green phase has been active for less than *maxGreenTime*, and can only choose *change* if the current green phase has been active for at least *minGreenTime*.

We remark that extending this scheme to cases in which the number of phases is higher than two is straightforward as each agent then has to select among a number of actions that correspond to the number of phases. Instead of the keep-change scheme just described, an agent just selects the phase number that will receive green light. This does not change the MDP formulation since the set of actions can have any size. For that, only the cardinality of the ρ set changes. However, experiments have to take into account that the agent potentially needs to experiment longer because the state space increases. This is likely to increase the necessary simulation horizon.

The agent keeps a record of each visited pair state-action, as well as an estimate of the Q-values for each of these pairs. Such values depend on the rewards obtained by performing action a at state s. Rewards are given by the number of vehicles that are queued in a given intersection. This value is provided by the microscopic simulator. As explained next, an agent may also receive information from other agents.

4.3. How Communication Works

Next we briefly explain how the communication is performed by the elements of the road network G. We assume that every junction $j \in J$ a is equipped with a communication device (henceforth, CommDev) that is able to send and receive messages to and from other junctions. For instance, in Fig. 2, junctions D2 and B2 have a relationship in VG, that refers to a given time step t. Thus, they initiate a communication via their respective communication devices in order for D2 to transfer its knowledge to B2.

This way, an agent that has received information perceives it as expected rewards for the action available to it in that particular state. This means that we extend the learning process vis-a-vis the standard Q-learning algorithm. When using Q-learning, at any given step t the agents simply update their Q-values based on the feedback from the action they have just taken at a given state. However, in our case, agents also update their Q-values based on the expected rewards received by the CommDev's. This means that at every time in which an agent j needs to make a decision, it asks and receives the following tuple regarding all agents with which it has a relationship in the VG: $< s_n, a_n, r_n >$, where n is a virtual neighbors of j. Each of these tuples is then used to update the Q-table of j (in case j has never seen a given pair (s, a), this pair is inserted in the table with the corresponding Q-value).

5. Experiments, Results, and Analysis

5.1. Scenario

Simulations were performed using SUMO [20], a microscopic simulator.

The test scenario is the traffic network with 12 intersections depicted in Fig. 2. Given that the links are one-way, not all intersections require a traffic signal controller. This is the case of the four corner intersections, as well as intersections A2, C1, and B3. Other non-signalized intersections are B1 and C3. These two are regulated by a priority mechanism defined in SUMO, namely all vehicles decelerate before reaching the intersection and SUMO regulates the right of using the intersection. We did this in order to concentrate on the intersections defined by the arterial depicted in the middle of the figure, namely the one that starts at intersection A2 and ends at D2. In this arterial, there are three signalized intersections that are then each controlled by a learning agent, namely B2, C2, and D2. Recall that, due to its geometry (no conflict between approach links), A2 does not require a traffic signal.

Each of the agents learns using the method described in Section 4.

We also stress that we have created this network with a particular feature in mind: the total number of vehicles is kept constant (except for the first steps, when they are still being inserted at the link B3-C3). This avoids many problems that are common in the literature, where it was not possible to fully isolate the behavior of the RL approach from SUMO's mechanism for routing vehicles.

In the network depicted in Fig. 2, there is a re-routing device in the link located right before intersection A2 that is responsible to re-route each vehicle when it reaches that link. Essentially, vehicles keep running in loops, never leaving the simulation. This allows us to control how the trips use the network, which is an important question we deal with here, as discussed ahead (non-stationarity). Specifically, 200 trips are generated (which, for this network represents almost 50% of overall density, given that its maximum capacity is around 500 vehicles).

Another feature of our scenario is non-stationarity due to changes in the traffic volume that traverses each intersection. In previous works such as [21, 16] it was shown that the respective methods were able to adapt to change in *traffic contexts* or simply *context*. By context we mean that, from time to time,

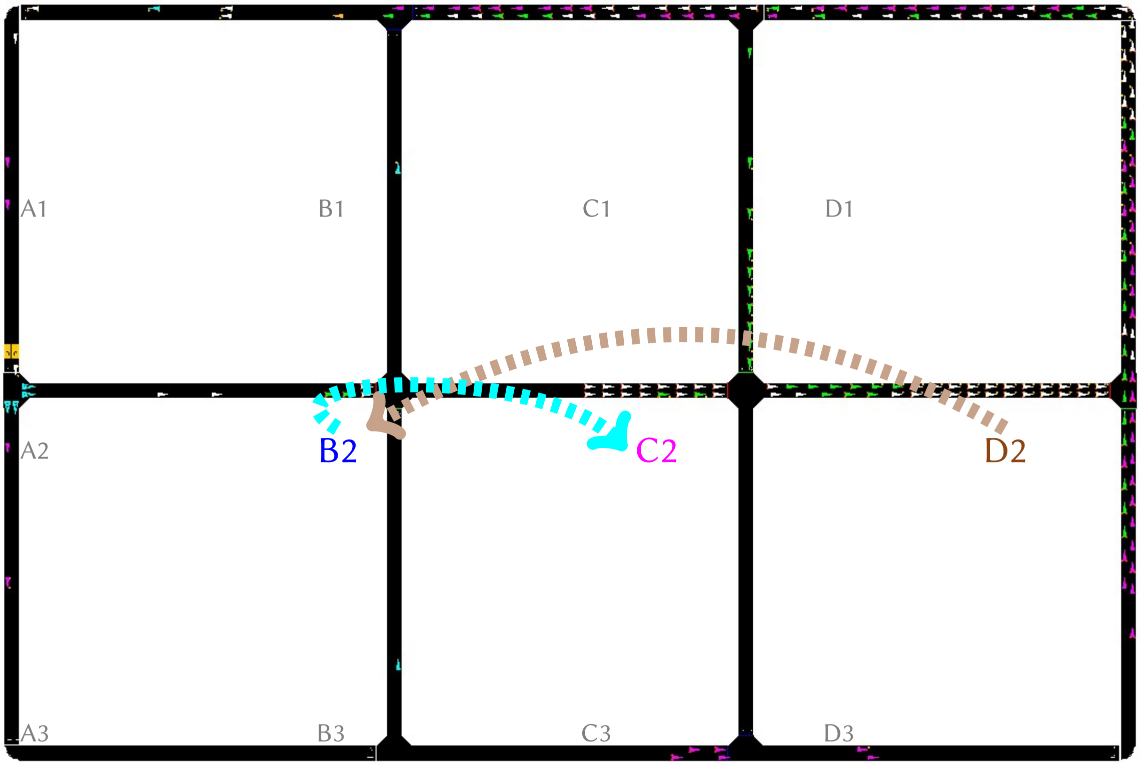


Figure 2: Road network used in the experiments. This figure also depicts an example of communication (D2 → B2 and B2 → C2).

the way trips are distributed over the routes are changed. Note that this does not mean that the number of trips (vehicles) using the network changes, but rather that each trip may use a different route, thus the use of the links, and, hence, of the intersections, differs along time.

These changes in context occur at each 5,000 simulation steps, as depicted in Fig. 3, where the time steps are represented in the y axis. Note that the figure just shows one change in context (at step 5,000), while in fact there is a change at each 5000 steps, in a repeated way.

As seen, in the first context (see bottom part of Fig. 3), both junctions B2 and C2 have similar situations – receiving more volume of vehicles in the horizontal direction – while D2 receives near equal volumes in both directions. At time step 5,000, there is a change in this patters (see top part of the figure). At time step 10,000 the former context occurs again and so on.

5.2. Values of the Parameters

Each simulation runs for 15,000 seconds. Our plots show the average and deviations over 15 repetitions of each case.

The value used for *minGreenTime* and for *maxGreenTime* were was 10 and 50 seconds respectively. Also, we recall that one action time step corresponds to five seconds of real-life clock time.

As for the learning parameters, after experimenting with other values, we have set $\alpha = 0.05$, $\gamma = 0.95$ and ε starting at 1.0 and decaying by the rate of 0.995 at each action time step, up to a minimum value of $\varepsilon = 0.05$, which is commonly used in the literature.

Finally, regarding the VG, we tried several values for the δ; here we show results for $\delta = 0.002$.

5.3. Results and Discussion

In order to assess the effectiveness and efficiency of our approach, we compare it to the case where Q-learning is used without the VG, as well as with the case in which no learning mechanism is used. Henceforth the latter is referred to as fixed time, since there is a fixed control rule, i.e., the timings

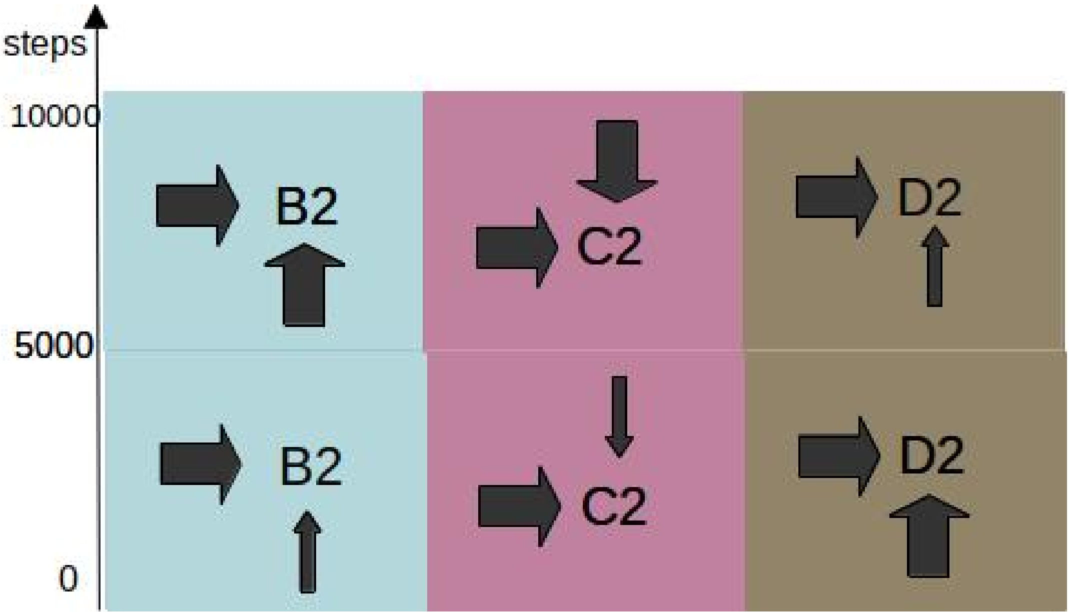


Figure 3: Changes of contexts along time steps.

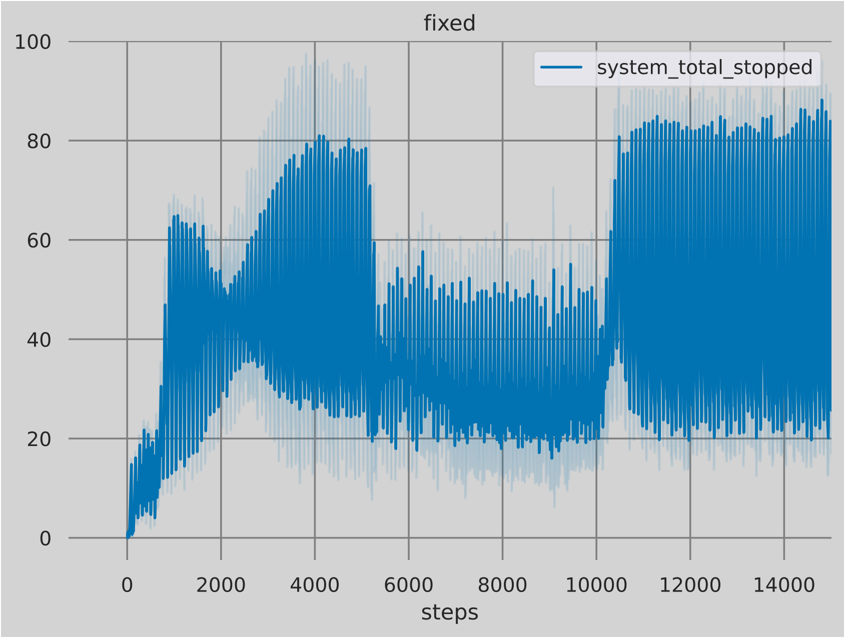


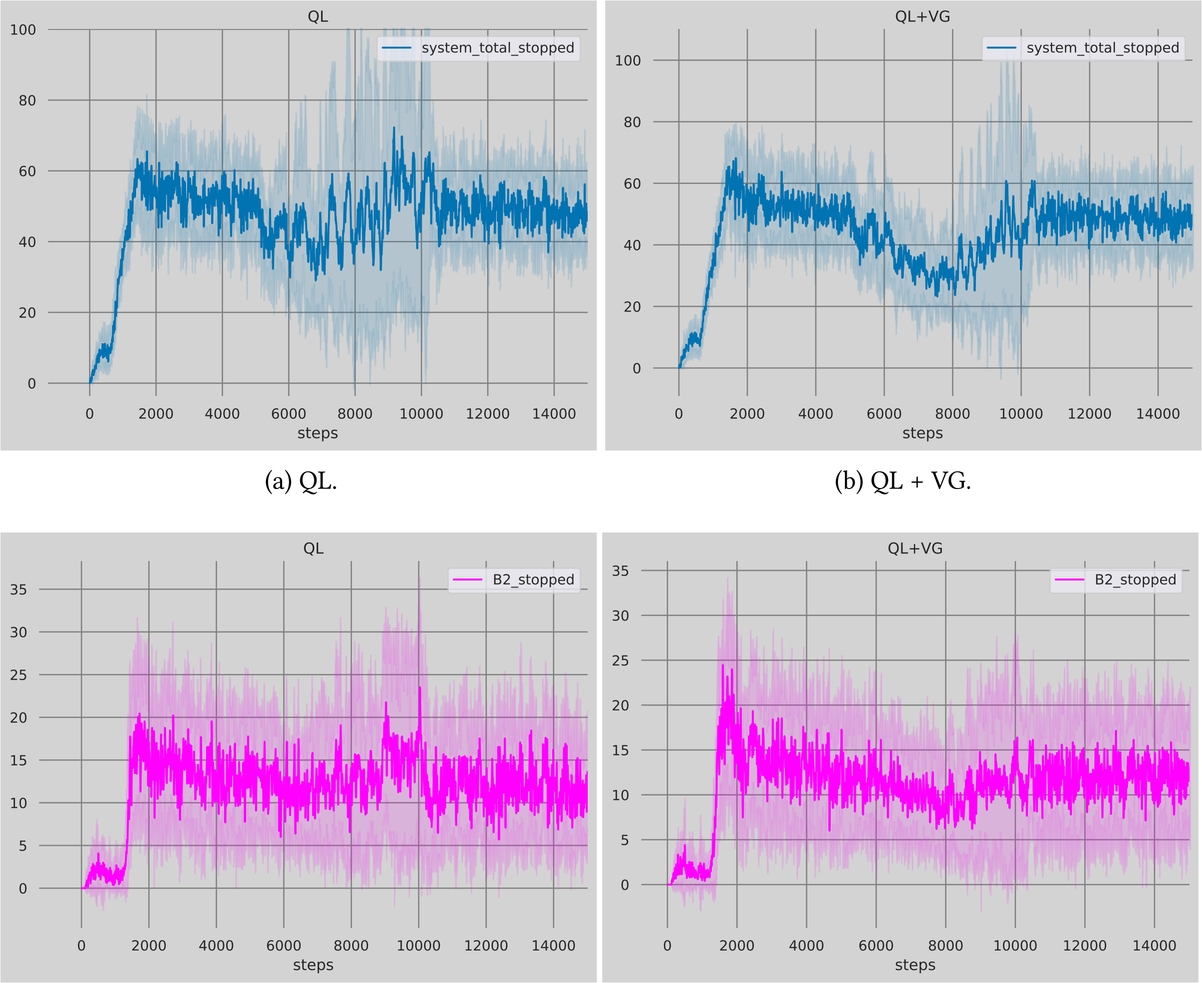
Figure 4: Number of stopped vehicles in the network: fixed time.

of the green signals are optimized once (by SUMO) and remain fixed. We remark that other types of controllers – not necessarily based on RL– could be employed for the sake of comparison; however, due to space limitations, we cannot discuss such results here.

In all cases, as mentioned, the network used is the one depicted in Fig. 2. The assessment is based on number of stopped vehicles, as commonly used in the literature.

Fig. 4 shows the number of stopped vehicles in the whole network (average over 15 repetitions), and it functions as a baseline, given that there is no learning mechanism of any kind. Rather, as aforementioned, the signal timings remain fixed. Note the oscillations found in the average curve (the deviations are given as a light color shadow), caused by the fact that vehicles are randomly routed, but the timing of the signals is fixed. Further, this is the case for all contexts.

We now show and discuss the cases in which the agents learn using QL or QL plus the VG.



(a) QL.

(b) QL + VG.

(c) QL (junction B2).

(d) QL + VG (junction B2).

Figure 5: Comparison of the number of stopped vehicles in the network and at two junctions.

Fig. 5 shows plots regarding Q-learning with (right side) and without (left side) the use of the VG. Recall that in both cases, there is a change in contexts each 5,000 time steps.

In the first line we show the total number of stopped vehicles (blue lines). Note the increase in performance, as compared to the case in Fig. 4, especially in the visible reduction in the oscillations observed in the 15 experiments.

We now compare both plots in the first line of Fig. 5, stressing what happens in the different contexts.

In the first 5,000 steps (first context), both approaches behave nearly the same because agents are still exploring (recall that we start with $\varepsilon = 1$ and decay this quantity). This combined with the fact that the patterns in the three intersections are either different (see discussion on Fig. 3) or are similar but the actions are not efficient for a given state (due to high/inefficient exploration), leads to no visible benefit of the use of the VG.

In the second context, the number of stopped vehicles in Fig. 5b is clearly lower than the one depicted in Fig. 5a, plus one can see that there is less oscillation and less deviation. In the third context, the performance is again nearly the same in both plots. This is due to the fact that this contexts repeats the first one; the agents have already learned and there is not so much new left to be shared among virtual neighbors.

The plots in the second line also show the number of stopped vehicles, but now just for intersection B2.

For this intersection, comparing Fig. 5c and Fig. 5d, it can be seen that Q-learning actually starts slightly better. This is due to the fact that agent at B2 receives information from the other two agents but, due to the initial exploration, such information may not be adequate, thus leading to the agent performing actions that are not efficient for its particular state. With time, this changes; here the best performance is seem within context number two, when B2 receives valuable information from the other agents.

For C2 and D2, there was no significant improvement. We are investigating two causes for this behavior. A first hypothesis is that C2 actually receives less traffic than the other intersections, in all contexts. This means that, even if the pattern is similar to B2, the magnitude of the flow is not the same. A second explanation may be related to the way the state space is coded, which could influence the way the information is communicated to other agents. Note that C2 has a vertical traffic direction that is opposite to B2, thus technically the states may be different.

As for D2, we only observe improvements in the third context, which means that this intersection is receiving noise information from the other agents. In a future work, we plan to filter out this noise by means of a smarter mechanism, able to put less weight on the received information, especially when it deviates too much from the already acquired one, i.e., the putting more weight on the local information.

In short, Fig. 5 shows that there is an advantage in letting virtual agents share their experiences. The performance improves to a different extent among the agents; some profit more than others.

6. Conclusion and future work

In this paper, we discuss how multiagent RL can be combined with transfer of experiences. The novelty of our approach lies in the fact that such transfer does not happen only among neighboring agents, but rather, among agents that have similar traffic patterns at a given time step. This is particularly useful when dealing with issues that arise when the state space is large. In such cases, a poor discretization may lead to poor performance.

We show the use of our approach using a traffic signal control scenario, where we also deal with changes in context, i.e., in the traffic flow patterns.

Our results show that, compared to a baseline stemming from standard RL, i.e., Q-learning, the proposed approach performs better in terms of stopped vehicle and also presents less oscillations.

We recall that we have considered emissions when constructing the virtual graph; however, we have used only travel time as reward. Thus, as a first next step, we plan to extend the approach to deal with more than one objective. Secondly, regarding the experiments, we intend to use scenarios in which agents are heterogeneous (e.g., they have a different set of actions that arise due to different set of phases), and investigate the issue of how states are coded and information transferred to other agents. Finally, we plan to extend the experiments in order to deal with situations where intersections require more than two phases, and also to perform experiments comparing our results to other kinds of traffic signal controllers.

Acknowledgements

This work is partially supported by CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil, Finance Code 001). Ana Bazzan is partially supported by CNPq under grant number 304932/2021-3.

References

[1] C. Watkins, Learning from Delayed Rewards, Ph.D. thesis, University of Cambridge, 1989.

[2] R. P. Roess, E. S. Prassas, W. R. McShane, Traffic Engineering, 3rd ed., Prentice Hall, 2004.

[3] A. L. C. Bazzan, Opportunities for multiagent systems and multiagent reinforcement learning in traffic control, Autonomous Agents and Multiagent Systems 18 (2009) 342–375. doi:10.1007/s10458-008-9062-9.

[4] P. Mannion, J. Duggan, E. Howley, An experimental review of reinforcement learning algorithms for adaptive traffic signal control, in: T. Leo McCluskey, A. Kotsialos, P. J. Müller, F. Klügl, O. Rana, R. Schumann (Eds.), Autonomic Road Transport Support Systems, Springer International Publishing, Cham, 2016, pp. 47–66. doi:10.1007/978-3-319-25808-9_4.

[5] M. Noaeen, A. Naik, L. Goodman, J. Crebo, T. Abrar, B. Far, Z. S. H. Abad, A. L. C. Bazzan, Reinforcement learning in urban network traffic signal control: A systematic literature review, 2021. URL: engrxiv.org/ewxrj. doi:10.31224/osf.io/ewxrj.

[6] H. Wei, G. Zheng, V. Gayah, Z. Li, Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation, SIGKDD Explor. Newsl. 22 (2021) 12–18. doi:10.1145/3447556.3447565.

[7] K.-L. A. Yau, J. Qadir, H. L. Khoo, M. H. Ling, P. Komisarczuk, A survey on reinforcement learning models and algorithms for traffic signal control, ACM Comput. Surv. 50 (2017). doi:10.1145/3068287.

[8] M. Aslani, M. S. Mesgari, M. Wiering, Adaptive traffic signal control with actor-critic methods in a real-world traffic network with different traffic disruption events, Transportation Research Part C: Emerging Technologies 85 (2017) 732–752. doi:https://doi.org/10.1016/j.trc.2017.09.020.

[9] P. Balaji, X. German, D. Srinivasan, Urban traffic signal control using reinforcement learning agents, IET Intelligent Transportation Systems 4 (2010) 177–188.

[10] S. El-Tantawy, B. Abdulhai, H. Abdelgawad, Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): Methodology and large-scale application on downtown toronto, Intelligent Transportation Systems, IEEE Transactions on 14 (2013) 1140–1150. doi:10.1109/TITS.2013.2255286.

[11] H. Wei, G. Zheng, H. Yao, Z. Li, IntelliLight: a reinforcement learning approach for intelligent traffic light control, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 2496–2505. doi:10.1145/3219819.3220096.

[12] M. Abdoos, N. Mozayani, A. L. Bazzan, Hierarchical control of traffic signals using Q-learning with tile coding, Appl. Intell. 40 (2014) 201–213. doi:10.1007/s10489-013-0455-3.

[13] E. Van Der Pol, Deep Reinforcement Learning for Coordination in Traffic Light Control, Ph.D. thesis, University of Amsterdam, 2016.

[14] L. N. Alegre, T. Ziemke, A. L. C. Bazzan, Using reinforcement learning to control traffic signals in a real-world scenario: an approach based on linear function approximation, IEEE Transactions on Intelligent Transportation Systems 23 (2022) 9126–9135. URL: ieeexplore.ieee.org/document/9468362. doi:10.1109/TITS.2021.3091014.

[15] A. L. C. Bazzan, D. de Oliveira, B. C. da Silva, Learning in groups of traffic signals, Eng. Applications of Art. Intelligence 23 (2010) 560–568. URL: http://www.sciencedirect.com/science/article/pii/S0952197609001699.

[16] V. N. de. Almeida, A. L. C. Bazzan, M. Abdoos, Multiagent reinforcement learning for traffic signal control: a k-nearest neighbors based approach, in: A. L. C. Bazzan, I. Dusparic, M. Lujak, G. Vizzari (Eds.), Twelfth International Workshop on Agents in Traffic and Transportation, volume 3173 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022, pp. 32–46. URL: http://ceur-ws.org/Vol-3173/3.pdf.

[17] A. L. Bazzan, H. U. Gobbi, G. D. dos Santos, More knowledge, more efficiency: Using non-local information on multiple traffic attributes, in: Proceedings of the X Symposium on Knowledge Discovery, Mining and Learning (KDMiLe 2022), SBC, Campinas, 2022, pp. 194–201. URL: https://sol.sbc.org.br/index.php/kdmile/article/view/24986. doi:10.5753/kdmile.2022.227737.

[18] A. L. Bazzan, H. U. Gobbi, G. D. dos Santos, Using non-local connections to augment knowledge and efficiency in multiagent reinforcement learning: an application to route choice, Journal of Information and Data Management 15 (2024) 186–195. doi:10.5753/jidm.2024.3328.

[19] H. U. Gobbi, G. D. dos Santos, A. L. C. Bazzan, Comparing reinforcement learning algorithms for a trip building task: a multi-objective approach using non-local information, Computer Science and Information Systems 21 (2024) 291–308. doi:10.2298/CSIS221210072G.

[20] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, E. Wießner, Microscopic traffic simulation using SUMO, in: The 21st IEEE International Conference on Intelligent Transportation Systems, 2018.

[21] L. N. Alegre, A. L. C. Bazzan, B. C. da Silva, Quantifying the impact of non-stationarity in reinforcement learning-based traffic signal control, PeerJ Computer Science 7 (2021) e575. URL: http://dx.doi.org/10.7717/peerj-cs.575. doi:10.7717/peerj-cs.575.