Computer Science and Information Systems 21(1):291–308 https://doi.org/10.2298/CSIS221210072G

Comparing Reinforcement Learning Algorithms for a Trip Building Task: a Multi-objective Approach Using Non-Local Information*

Henrique U. Gobbi, Guilherme Dytz dos Santos, and Ana L. C. Bazzan

Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS)
Porto Alegre, RS, Brazil
{hugobbi,gdsantos,bazzan}@inf.ufrgs.br

Abstract. Using reinforcement learning (RL) to support agents in making decisions that consider more than one objective poses challenges. We formulate the problem of multiple agents learning how to travel from A to B as a reinforcement learning task modeled as a stochastic game, in which we take into account: (i) more than one objective, (ii) non-stationarity, (iii) communication of local and non-local information among the various actors. We use and compare RL algorithms, both for the single objective (Q-learning), as well as for multiple objectives (Pareto Q-learning), with and without non-local communication. We evaluate these methods in a scenario in which hundreds of agents have to learn how to travel from their origins to their destinations, aiming at minimizing their travel times, as well as the carbon monoxide vehicles emit. Results show that the use of non-local communication reduces both travel time and emissions.

Keywords: reinforcement learning, multi-agent systems, multi-objective reinforcement learning, route choice.

1. Introduction

Recent publications in the area of multi-agent systems are showing the value of using multi-objective reinforcement learning (RL) in agents' decision making processes. The rationale here is that many tasks are better dealt with when multiple – possibly conflicting objectives – are considered. Although this poses more challenges to RL, especially when more than few agents interact, such formulation is useful in real-world problems as for instance making decision regarding trips in traffic networks. In this domain, multiple drivers must learn to reach their destinations, starting at their given origins. Depending on the formulation of the RL task, agents construct their routes by making decisions about which link to follow, once they find themselves at given locations. Thus, locations are states and actions are selection of links. This way, agents learn how to construct a route from their origins to their destinations.

Usually, for such learning task, a single objective is considered, namely minimizing travel times. However, frequently, there are other objectives to be considered. For instance, there are works that optimize two objectives – toll and travel time – such as [14, 21, 22]. These works are based on methods that are centralized and do not involve RL.

* Extended version of a paper presented at the ATT 2022 workshop.

Route choice using travel time and toll by means of RL is addressed by [8]; however this work deals with agents selecting among k pre-computed routes that take the agents from their origins to their destinations. This means that the RL task involves only one state (stateless RL), namely the origin node, where an agent makes a decision (select one of the k routes). Then the agent follows the selected route without making further decisions during the trip. For this kind of problem, the work described in [8] extends a Bandit algorithm like UCB [1], in order to account for multiple objectives and for multiple learning agents.

In contrast to the aforementioned works, in the present paper, each agent or driver not only performs its optimization process locally (in a decentralized way), by means of multi-objective RL (MORL), but also it builds its trip by making decisions at each intersection (that function as states). Hence, the underlying learning task is state-based.

Other characteristics of the problem we deal with is that it involves many agents competing for resources. This also poses challenges to RL methods, even if only one objective is considered. Specifically, the fact that there are many agents learning simultaneously causes the environment to be non-stationary.

As these features make the learning task hard, in this paper we aim at investigating the benefits of communication among the various actors (vehicles, elements of the road infrastructure) and, in particular, the role of using non-local information. The rationale for this research question relates to benefits reported in the area of new communication technologies such as vehicle to infrastructure (V2I) communication ([9, 10, 17–19]). We posit that, by effectively using communication-based approaches, congestion and, consequently, emissions can be reduced.

In short, a decentralized RL-based approach to routing involves the following issues: (i) agents learning simultaneously result in non-stationarity; (ii) there are potentially two or more objectives to be optimized; (iii) the underlying learning task is modeled as a stochastic game, where states (in which decisions about which link to take are made) are the intersections of the networks; (iv) communication among the various actors in the road network.

The remainder of this paper is organized as follows. The next section discusses background concepts and gives an overview on the related work. Section 3 details the proposed method. Its evaluation in a proof-of-concept scenario is discussed in Section 4. Concluding remarks and future directions are given in Section 5.

2. Background and Related Work

In this section, we briefly introduce underlying concepts on RL, as well as on traffic assignment, route choice, and routing (including multi-objective variants), as well as on V2I and other types of communication that are possible in a road network.

2.1. Reinforcement Learning

In RL, an agent learns how to act in an environment, by receiving a feedback (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, which

characterizes an exploration phase. Such phase may be very noisy in multiagent scenarios, given that all agents are learning simultaneously and hence have to adapt to others' exploration processes. However, agents should not only explore; in order to maximize the rewards of their actions, they also have to exploit the gained knowledge. Thus, there must be an exploration-exploitation strategy that is to be followed by the agents. 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).

In the exploitation phase, at each interaction, it is assumed that an agent has sensors to determine its current state and can then make an action. The reward perceived or received from the environment is 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 paper, we concentrate on a model-free, off-policy algorithm called Q-learning or QL [23] and its extensions. In QL, 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.

The reason for using QL is based on the fact that it has proven both effective and efficient, i.e., leads to agents learning routes that quickly optimize some quantity (normally, but not only, travel time). See references in the next section.

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

RL can be employed to compute how drivers learn how to go from A to B in a transportation network. The next section thus introduces this learning task.

2.2. How to Travel from A to B?

In this section, we first discuss the terminology that is related to this problem. Then, we introduce concepts underlying the definition of an urban network (topology), as well as the demand side. Finally, we discuss the need for multi-objective approaches.

Given a road network and the demand for its use (number of trips per unit of time), the task of finding out how each unit of a demand realizes its travel needs can be solved in different ways, although most of them seek to compute the so-called user equilibrium, i.e., the condition in which no user is better off by changing its route. We recall that, in the literature, demand can be expressed as user, trip, traveller, agent, or vehicle, depending on the community.

In the transportation community this problem is normally known as the traffic assignment problem (TAP), which is solved in a centralized fashion, mostly via a macroscopic approach using a volume-delay function (VDF), which estimates the travel time as a function of the volume of trips; see Chapter 10 of [12] for details. In the computer science community, that task is frequently solved by means of RL in a decentralized way, in the sense that agents learn how to reach their destinations, by performing experiments (exploration) until they learn their respective routes (which is equivalent to the condition under the user equilibrium). This tasks admits two variants: in the first, an agent knows a set of precomputed routes (or is able to compute them), and its task is to choose a route among them (again by means of experimenting with several selections of actions

or routes). Once such choice is done, an agent travels the route, without changes during the trip. Examples of this approach are [11, 15] among others. The second variant departs from the assumption that agents are given a set of k shortest routes. Instead, agents make choices during the trip (e.g., at each intersection, or at each important point of decision), thus building the actual route while traveling, i.e., they decide which link to follow once they find themselves in each vertex. This is the approach followed in [2, 19] and also in the present paper.

In terms of RL, these two variants have a key difference. While the latter is a classical RL task with a set of states (e.g., each decision points), and a set of actions, the former belongs to the stateless RL front, where agents basically need to select an action at their respective origin node or state. Moreover, this learning task resembles Bandit algorithms such as UCB [1]. Henceforth, we refer to the former as route choice (as the agent simply chooses a route and remain on it), while the latter is referred here as routing to convey the idea that the trip is built during the actual routing task.

Regarding approaches that were already proposed to tackle this problem, in the literature, there are various approaches to solve the TAP; here we refer only to those that address more than one objective such as [5, 14, 21, 22]. Others can be found in [3].

Regarding route choice, the reader is referred to [11, 15]; in particular, multi-objective route choice is tackled by [8]. All these works use a macroscopic approach in the sense that VDFs are used to provide a reward for each agent based on the flow on a given route.

Routing is less common in the literature but some works appear in [6, 17–19]. While the former employs a VDF to compute rewards (thus, a macroscopic approach), the last two use a microscopic simulator, where vehicles realize the actual driving movements. As such, rewards (e.g., travel times) are given by the simulator itself and correspond to a more realistic reward (as opposed to the estimation provided by a VDF).

Next we briefly review the concept of transportation network and the demand that uses it. Formally, a traffic network can be modeled as a graph $G = (V, L)$, where V is the set of vertices or nodes that represent the intersections, and L, the set of links that describe the segments connecting a pair of vertices. In the network, trips are distributed among a set of origin-destination (OD) pairs; each corresponding to a certain demand for trips. These then translates into flows on the various links.

In all methods it is assumed that agents (drivers) are rational, thus each selects the route or link with the least perceived individual cost, in order to travel from its origin to its destination. Several factors may influence this decision, such as travel time, distance, monetary cost (e.g., toll), fuel consumption, battery, emission of pollutants, etc.

Traditionally, multi-objective traffic assignment is modeled by using a linear combination of the various objectives, as proposed, for instance, by Dial in [5] for a bi-objective assignment. Such a linear combination has some drawbacks. For instance, efficient routes may be missed, as discussed in [21]: in a nutshell, the key point is that algorithms that use a linear combination only identify supported solutions at extreme points, while there may be other efficient solutions that are not considered in that group but could be preferred by some users.

Hence, it is necessary to have alternative solutions. One issue that arises is that, in a multi-objective scenario, there is a set of efficient solutions rather than just one. Different solutions have been proposed in the transportation and optimization communities; see for instance [5, 14, 21, 22]. We remark that all these methods rely on a set of (pre-computed) k

shortest paths and are centralized. Thus, they are not useful in a multiagent environment where each agent should learn by its own experience using RL, nor they fit a state-based RL task, where agents learn at each intersection.

2.3. A Multi-objective Approach to Q-learning

In this section we discuss how to use QL when agents need to deal with more than one objective. For more details we refer readers to other sources, which also cover other approaches; see [7, 13].

In order to extend the aformentioned QL algorithm, [20] proposed Pareto Q-learning (PQL), which integrates the Pareto dominance relation into a MORL approach. PQL considers both the immediate reward vector and the set of expected future discounted reward vectors in order to compute the so-called Q-sets, which are composed of vectors. In PQL, the set of expected future discounted reward vectors relies on a function, called ND (from non-dominated), that finds those vectors that correspond to the possible future states, and which are not Pareto-dominated.

Eq. 2 shows how Q-sets are calculated. $\bar{R}(s,a)$ denotes the immediate reward vector and $ND_t(s,a)$ is the set of non-dominated vectors in the next state s, which is reached by performing action a at time step t. $\bar{R}(s,a)$ is added to each element of $\gamma\, ND_t(s,a)$. When a is selected at s, both terms are updated. $\bar{R}(s,a)$ is updated according to Eq. 3, where $\vec{R}$ is the new reward vector and $N(s,a)$ is the number of times action a was selected in s. $ND_t(s,a)$ is updated as shown in Eq. 4, using the non-dominated vectors in the $\tilde{Q}_{set}$ of every action a' in the next state s'.

$$\tilde{Q}_{set}(s,a) = \bar{R}(s,a) \oplus \gamma\, ND_t(s,a) \tag{2}$$

$$\bar{R}(s,a) = \bar{R}(s,a) + \frac{\vec{R} - \bar{R}(s,a)}{N(s,a)} \tag{3}$$

$$ND_t(s,a) = ND(\cup_{a'} \tilde{Q}_{set}(s',a')) \tag{4}$$

PQL learns the entire Pareto front, finding multiple Pareto optimal solutions, provided that each state-action pair is sufficiently sampled. This algorithm is not biased by the Pareto front shape (algorithms that find a single policy and use scalarization cannot sample the entire Pareto front if it is non-convex) or a weight vector (it guides the exploration to specific parts of the search space).

Note that PQL assumes that the environment is deterministic, hence it is not directly useful for environments in which the presence of multiple agents learning simultaneously causes non-stationarity, as for instance the route choice domain we discuss ahead. Therefore, we adapted PQL to deal with multiple objectives and with multiple agents (thus, a non-stationary environment). Henceforth this adapted algorithm is denoted as aPQL.

2.4. Communication in Road Networks

As aforementioned, communication in road networks (among vehicles, between vehicles and infrastructure, etc.) has started to receive attention also in the traffic engineering community. The reader is referred to [9] (focusing on how autonomous vehicles and connected

vehicles are expected to increase the throughput of highway facilities, as well as improve the stability of the traffic stream), and to [10] (for applications).

The use of this kind of technology was investigated by us previously in [17–19], where we have connected it to MARL, in order to investigate how V2I communication could help drivers in RL-based trip building or routing. In these works, the infrastructure is able to communicate with the vehicles, both collecting information about their most recent travel times (on given links), as well as providing them with information that was collected from other vehicles. However, besides considering a single objective only, links in the infrastructure only exchange information if they are connected by a junction, i.e., only local information is considered, whereas in the present paper we also consider that those links share non-local information that they acquire from links that are similar to them. For this, as detailed ahead, we employ a virtual graph where nodes are the (road) links and edges exist between any two links that have similar values in terms of travel time and emission. This kind of approach was also used by [4]. However, that work left the use of multiple objectives as an open direction.

3. Methodology

This section details our methods. We start by defining the virtual graph that connects elements of the road network, thus allowing them to communicate and exchange information that will then be passed to drivers, to help these to make decisions. These are the topics presented in the next two subsections. We then formalize the learning task itself, ending the section recapitulating the main points of the approach.

3.1. Terminology: Road Network and Virtual Graph

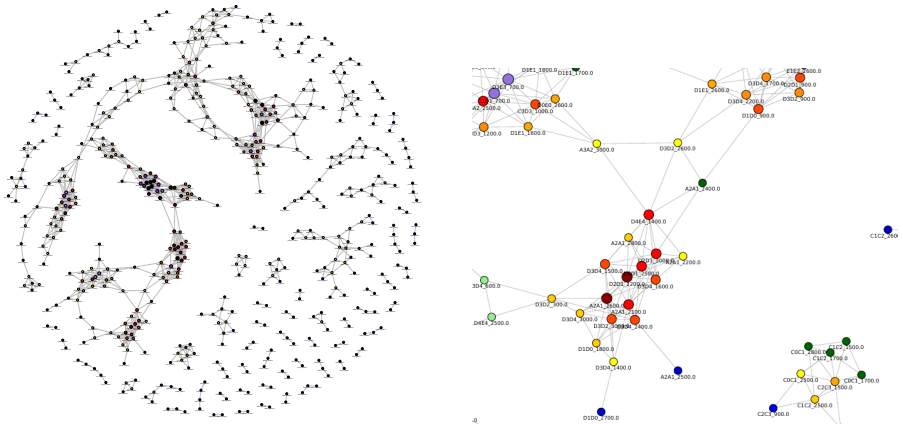
We start by stressing that we deal with two sorts of graphs. First, a road network is a (planar) graph $G = (I, L)$, where I is the set of intersections, and L is the set of links. In short, G represents the road network and mirrors its topology. For example, as discussed ahead in more details, in Fig. 2 G is a 5×5 grid, in which intersections `A4` and `B4` are connected by a link `A4<-->B4`.

The other graph refers to non-local connections among two (not necessarily physically close) links $l_1 \in L$ and $l_2 \in L$ (as, e.g., `A4<-->B4` and `C1<-->D1` in Fig. 2). A connection in this graph arises if both show similar patterns. We call this a virtual graph denoted by $VG = (L, E)$, where L is the set of links (note that now links act as vertices in VG), and E is the set of edges that connect two links that have similar patterns, as described next in more details.

In order to apply non-local communication to share information across the network, VG connects links in the network that share similar attributes. In VG, every node is a link $l \in L$ at a specific time step. An edge is only created when two of these nodes share similar attributes. This happens as follows: first of all, data related to some attributes of the links is gathered along time. This data can either be historical data or be collected in a simulation. This way, various attributes, such as travel time, link occupancy and CO (carbon monoxide) emission are collected at each time step. The data is then normalized and used to compare every link at a time step against every other link, checking if the absolute difference between the values of their attributes is within a defined threshold Δ.

If two links have attributes whose values are within Δ, an edge between them is added to VG.

Fig. 1a shows an instance[1] of such a virtual graph (corresponding to the network G depicted in Fig. 2), whereas, for a better visualization, Fig. 1b shows a zoom of that graph, where some relationships among similar links can be better seen. The labels of the vertices are formed by the link ID plus the time interval in which their values were found to be similar.



(a) Full VG **(b)** Zoom of a small part of the VG

Fig. 1. Instance of a virtual graph VG

3.2. Communication Using the Virtual Graph

The central idea of the communication using the virtual graph is to extend the communication between driver agents and elements of the road infrastructure (mostly the links via communication devices installed on them), so that such elements send information about the status of the links to drivers. For example, assume that l_1 and l_2 are neighbors in the VG because, despite not being close, they share similar values for some selected attributes. This means that they both exchange information, which is then passed to those driver agents that intend to use such links.

More specifically, an intersection collects information from its incoming links, defined in the physical road network G. Additionally, given that these incoming links may have virtual neighbors in the virtual graph VG, information about their virtual neighbors are also passed to the links. Once an intersection i has collected such information, it updates a table in which the last 30 entries are kept in a FIFO way, for each attribute. This window size was used in [18] for the same scenario. In the present paper, an intersection stores information also about travel time and CO emission.

[1] For sake of clarity, we have generated such figure using an arbitrary low number of time steps; the actual graph is denser than represented in that figure.

The intersection then communicates to each nearby driver agent an aggregation of those values kept in the tables, i.e., potential rewards that the driver may obtain if selecting each action in that particular state. The agent then perceives this information as expected rewards for the actions available to it.

To summarize, the purpose of the virtual graph VG is to indicate which links in the road network should exchange information about the network during communication at various moments throughout the simulation. The VG relates links that share similar patterns, therefore these links will exchange information with each other during the simulation in an effort to accelerate and improve the learning of the agents, given that the experiences of the agents are widely shared across the network.

3.3. The Learning Task Formalized as a Markov Decision Process

In this section, we discuss how the learning task was formalized as an MDP (see Section 2.1).

Recall that driver agents build their routes by making decisions about which link to follow, when finding themselves in a given intersection. Thus, in this routing learning task, each intersection $i \in I$ within a traffic network G is a state s, and the actions correspond to the links an agent might take when in s (i.e., links that leave the intersection).

Regarding the environment, the transition model is implemented by a microscopic traffic simulator, which moves vehicles along the network.

As for the rewards, since we deal with a multi-objective scenario, the rewards returned by the simulator are assembled in a vector; this way, the reward function definition is $R : S \times A \rightarrow \mathbb{R}^n$, where n is the number of objectives. As presented ahead in Section 4.2, in the present paper we deal with agents aiming at optimizing two objectives: travel time and carbon monoxide emission. However, the proposed method is general and can be used if more objectives are taken into account.

At this stage, a note about action selection is necessary. Originally, to deal with reward being a vector, in the PQL algorithm (i.e., as proposed in [20]), every pair state-action (s, a) is associated with a set of non-dominated points that represents the Pareto front. Q-sets are computed as in Eq. 2 and are used to compute a hypervolume of the Pareto front. This is then used to determine the best action. In our case, this was not effective given that the Q-sets tend to contain few points only, thus computations based on hypervolume tend to be ineffective. Therefore, we had to modify the action selection strategy. Instead of using the hypervolume of the Pareto front to determine the best action, we let each agent select randomly (with uniform probability, at each decision point, and independently of one another) which objective is to be optimized at that given step. This seems to be a fair substitution for the hypervolume and avoids biasing towards one objective. Once a particular objective is randomly drawn, then an agent uses the ε-greedy strategy to select an action, given the Q-set. Note that the approach continues to be multi-objective, as all objectives have their values updated after a given action was selected. To differentiate from PQL, we refer to our approach as aPQL.

3.4. The Whole Picture

Algorithm 1 shows how our method works for each agent, in terms of how it deals with learning when getting additional information gain from communication. Each agent trav-

Algorithm 1: Main Procedure (for each agent)

```
   Data: M, origin, destination, learningParameters
1  step ← 0
2  agent starts trip in its origin
3  while step < M do
4      if agent has reached a decision point then
5          r⃗ ← reward from last link
6          communication agent and infrastructure
7          computation of potential rewards
8          update of knowledge base
9          if reached its destination then
10             agent restarts trip in its origin
11         else
12             agent chooses a new action (link to travel)
13         end
14     end
15     step ← step + 1
16 end
```

els the links and make decisions at the intersections, until $step = M$ (maximum number of steps).

Lines 4 through 14 correspond to the behaviour an agent has when it reaches an intersection. It perceives its reward from traveling through a link (Line 5): this reward is a vector composed by one element per objective (e.g., two objectives for travel time and CO emission). The agent then communicates with the intersection's infrastructure element (IE), informing its perceived reward and retrieving the expected rewards, as discussed in Section 3.2.

It then uses its own perceived reward, as well as the expected rewards communicated by the IE (Line 8) in order to update its knowledge base. For our experiments (Section 4), we compare QL with aPQL. An agent learning using QL thus updates its Q-Table (Eq. 1), while an agent using aPQL updates its ND set (Eq. 4).

If an agent reaches its destination, it gets reinserted at its corresponding origin to start a new trip. Otherwise, it chooses which link to travel next. This action selection uses the $\varepsilon - greedy$ strategy, as discussed in Section 3.3. This means the agent will take the action which maximizes its future gains with a probability of $(1 - \varepsilon)$, or take a random action with a probability of ε.

4. Experiments and Results

4.1. Network and Demand

To test the approach, we used a 5×5 grid network depicted in Fig. 2. The demand corresponding to the OD (origin-destination) matrix is shown in in Table 1. Each link shown in Figure 2 is two-way. We recall that both the network and this demand were used in [16]. Simulations were carried out using a microscopic traffic simulator, namely SUMO.

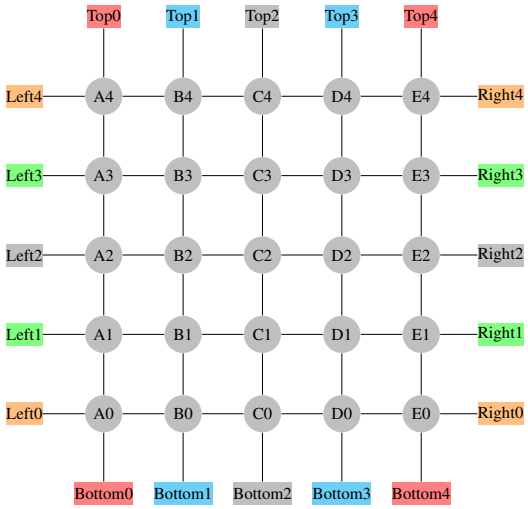


Fig. 2. 5x5 Grid Network (as in [19])

4.2. Specifying the MDP For This Scenario

In a traditional RL environment, at each time step, each agent finds itself in state s, chooses action a, receives a reward and transitions to a state s'. In the learning task at hand, time steps are controlled by the microscopic simulator and correspond to a unit of time (such as one second). This has important consequences. First, not all time steps count as decision-making steps; these only happen when an agent is at an intersection. The rest of the time steps are used (e.g., see plots ahead) just as clock units. Second, different agents find themselves either in decision-making states, or are moving along a link (thus, not updating their value functions), or have reached their destinations. In the latter case, an episode is finished *for that particular agent*. Since travel times and destinations are different for most of the agents, episodes are *not synchronous*. Therefore, in general, we do not refer to episodes; rather, we use the clock units or time steps of the simulator to depict time. To account for the simulation time, we use the parameter M (maximum number of steps), as introduced in Section 3.

Next, we detailed how we deal with multiple objectives in the reward function for this specific scenario.

In the exploration phase, due to uncoordinated action selection by the agents, congestion may occur in the road network, which impacts the reward of the agents, while also inserting noise in the learning process, which then leads to slow convergence. In order to speed up this process, and avoid agents wandering around the road network performing experimentation, two hyper-parameters were used. See [16] for details.

4.3. Parameters and Their Values

The parameters discussed in Section 3 were set as follows: maximum simulation steps $M = 40,000$ and threshold $\Delta = 0.005$. Two attributes were considered, namely travel

Table 1. Demand (nb. of trips) per OD-pair

OD-Pairs	Demand
Bottom0—Top4	51
Top4—Bottom0	51
Bottom4—Top0	51
Top0—Bottom4	51
Bottom1—Top3	43
Bottom3—Top1	43
Top3—Bottom1	43
Top1—Bottom3	43
Left1—Right3	43
Right3—Left1	43
Left3—Right1	43
Right1—Left3	43
Left0—Right4	51
Right4—Left0	51
Left4—Right0	51
Right0—Left4	51

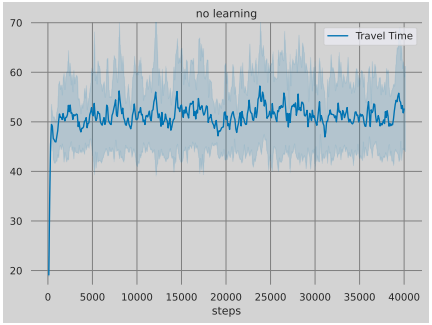
time and CO emission. We remark that these values are provided by the traffic simulator (SUMO).

The method proposed was compared against: no learning (which means that the driver agents follow routes given by SUMO's trip assignment); QL without the use of the virtual graph VG; QL combined with VG; aPQL without VG; and aPQL combined with VG. Recall that, when QL was used, the reward was a scalar, i.e., either travel time or CO emission (but not both together as it is the case with aPQL). Hence, we have run two distinct simulations: one in which the reward is based on the travel time alone, and another in which the reward is given by the CO emission. Following values reported in [16], for QL we have set the parameters – for both QL and aPQL– as follows: learning rate $\alpha = 0.5$, discount factor $\gamma = 0.9$, and $\varepsilon = 0.05$.

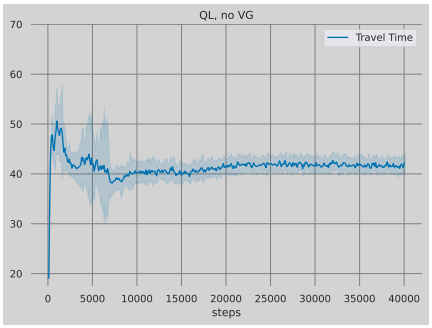
Its is also worth mentioning that, due to the stochastic nature of all methods we used for comparison, each of them were repeated 10 times. In the plots ahead, we show the mean values (and their standard deviations, as shadows).

4.4. Discussion of the Results

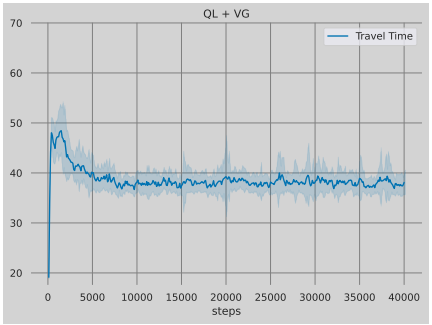
We first discuss plots that relate to travel time. For the sake of showing some baselines, in a first group (Fig. 3), we compare plots regarding QL with and without the use of the VG, and these against the case without learning. The latter is shown in Fig. 3a, where we can observe an average travel time per link around 50 time steps (or seconds). Note also that this plot is associated with a high deviation over the 10 repetitions.



(a) No learning



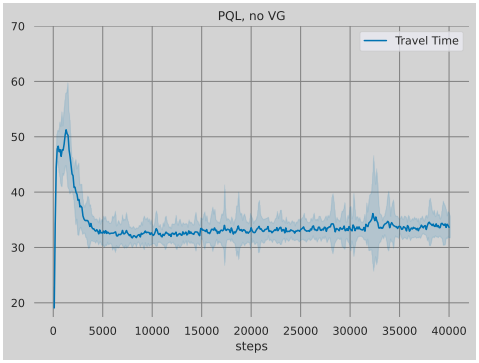
(b) QL



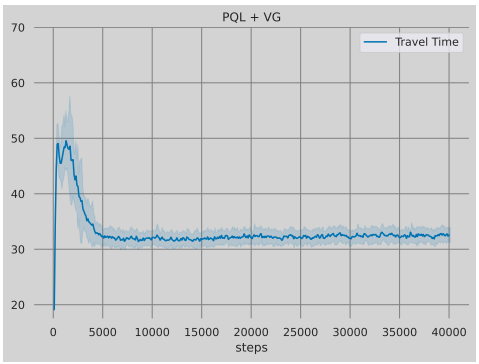
(c) QL plus virtual graph

Fig. 3. Travel time (average per link)

The cases in which QL was used appear in Fig. 3b and Fig. 3c. We can see that in the former, there is an improvement in performance (compared to the case without learning), as the travel time decreases to a value slightly over 40 when QL is used without the VG. When the VG is used, we see a further improvement. We stress that in both these cases, only travel time is being optimized so the use of the virtual graph is of reduced value.



(a) aPQL



(b) aPQL plus virtual graph

Fig. 4. Travel time (continued)

What interests us is indeed the case in which both objectives are considered together, i.e., the multi-objective approach that is proposed here. This case is shown in two plots in Fig. 4. Fig. 4a shows results for aPQL without the use of the VG, while Fig. 4b depicts the case in which VG is used. In both cases we see an improvement when compared with plots in Fig. 3: now the travel time converges to a value slightly over 30, with the latter – in which the VG was used – being better, as the travel time is lower and there are less oscillations.

It may not be completely obvious why in the multi-objective case we also observe a reduction in the travel time. We have conducted a (not yet extensive) investigation and concluded that the reason for such behavior is that, by minimizing emissions together with travel time, certain routes are chosen that distribute the vehicles better in the network

(as this reduces emissions). This in turn leads to lesser travel times. It remains to be investigated how correlated these two objectives are.

Similarly, Fig. 5 shows the results that relate to emission of CO. The plot in Fig. 5a shows the baseline when no learning algorithm was used. One observes a convergence to an emission of around 600 in mg/s. This value is then decreased when QL is used: to around 500 (Fig. 5b). When QL is combined with the VG (Fig. 5c), there is a further improvement.

Then, in Fig. 6, we show the plots for the multi-objective approach, where we can observe an even higher improvement, with the emission decreasing to values over 400. Finally, when the VG is combined with aPQL, the values decrease to below 400.

5. Conclusion and Future Work

Using RL to support agents making decisions that consider more than one objective poses challenges. We formulate the problem of multiple agents learning to travel from A to B in a traffic network as an RL task in which: (i) agents learning simultaneously result in non-stationarity; (ii) there are potentially two or more objectives to be optimized; (iii) the underlying learning task is modeled as a stochastic game, where states are intermediary locations in the networks (such as intersections), in which decisions are made about which link to take; (iv) non-local information is shared among elements of the infra-structure, which is then passed to driver agents.

To solve this task, we propose an adaptation in an existing algorithm, PQL [20]. This algorithm deals with more than one objective, but it was originally formulated for single-agent learning tasks, in which the environment is deterministic. In route choice, neither of these apply. Thus our adaptation addresses them. Moreover, we have combined this algorithm with a virtual graph that allows non-local communication.

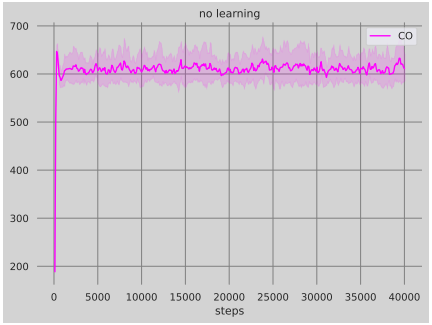
We have applied the proposed method to a scenario in which driver agents compete for a scarce resource (link usage), and have to learn how to travel from their origins to their destinations, aiming at minimizing their travel times, as well as the CO vehicles emit. Results show that the adapted algorithm, combined with the virtual graph is able to find routes that result both on lower travel times as well as emissions of CO.

As aforementioned, we intend to investigate the effect of a possible correlation between the two objectives we are using in the learning task.

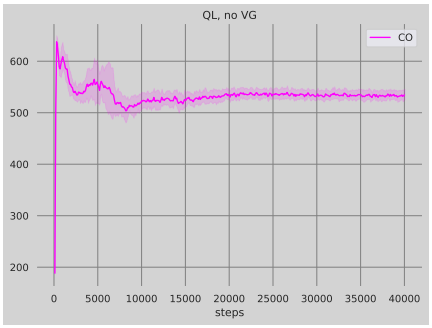
Acknowledgments. Ana Bazzan is partially supported by CNPq (grant 304932/2021-3). This work was partially funded by FAPESP and MCTI/CGI (grants number 2020/05165-1) and by CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil, Finance Code 001), and also partially sponsored by the German Federal Ministry of Education and Research (BMBF), Käte Hamburger Kolleg Cultures des Forschens/ Cultures of Research.

References

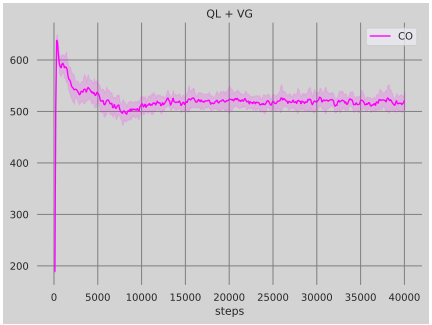
1. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning 47(2/3), 235–256 (2002)
2. Bazzan, A.L.C., Grunitzki, R.: A multiagent reinforcement learning approach to en-route trip building. In: 2016 International Joint Conference on Neural Networks (IJCNN). pp. 5288–5295 (July 2016)



(a) No learning

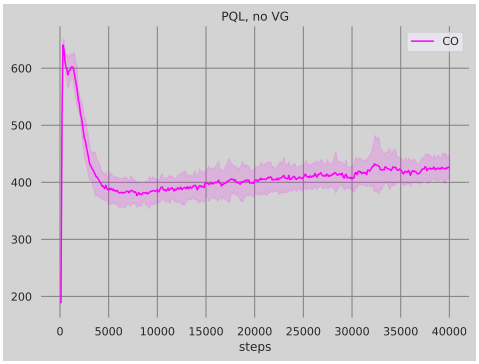


(b) QL

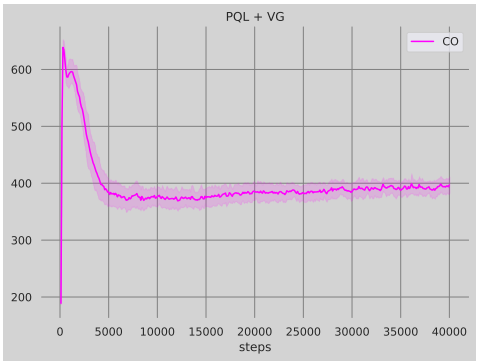


(c) QL plus virtual graph

Fig. 5. CO emission in mg/s (average per link)



(a) aPQL.



(b) aPQL plus virtual graph.

Fig. 6. CO emission in mg/s (continued)

3. Bazzan, A.L.C., Klügl, F.: Introduction to Intelligent Systems in Traffic and Transportation, Synthesis Lectures on Artificial Intelligence and Machine Learning, vol. 7. Morgan and Claypool (2013)
4. Bazzan, A.L., Gobbi, H.U., dos Santos, G.D.: More knowledge, more efficiency: Using nonlocal information on multiple traffic attributes. In: Proceedings of the KDMiLe 2022. SBC, Campinas (November 2022)
5. Dial, R.B.: A model and algorithm for multicriteria route-mode choice. Transportation Research Part B: Methodological 13(4), 311–316 (1979)
6. Grunitzki, R., Bazzan, A.L.C.: Combining car-to-infrastructure communication and multiagent reinforcement learning in route choice. In: Bazzan, A.L.C., Klügl, F., Ossowski, S., Vizzari, G. (eds.) Proceedings of the Ninth Workshop on Agents in Traffic and Transportation (ATT-2016). CEUR Workshop Proceedings, vol. 1678. CEUR-WS.org, New York (July 2016), http://ceur-ws.org/Vol-1678/paper12.pdf
7. Hayes, C.F., Radulescu, R., Bargiacchi, E., Källström, J., Macfarlane, M., Reymond, M., Verstraeten, T., Zintgraf, L.M., Dazeley, R., Heintz, F., Howley, E., Irissappane, A.A., Mannion, P., Nowé, A., de Oliveira Ramos, G., Restelli, M., Vamplew, P., Roijers, D.M.: A practical guide to multi-objective reinforcement learning and planning. CoRR abs/2103.09568 (2021), https://arxiv.org/abs/2103.09568
8. Huanca-Anquise, C.A.: Multi-objective reinforcement learning methods for action selection: dealing with multiple objectives and non-stationarity. Master's thesis, Instituto de Informática, UFRGS, Porto Alegre, Brazil (2021), http://hdl.handle.net/10183/231836
9. Mahmassani, H.S.: Autonomous vehicles and connected vehicle systems: Flow and operations considerations. Transp. Sci. 50(4), 1140–1162 (2016)
10. Maimaris, A., Papageorgiou, G.: A review of intelligent transportation systems from a communications technology perspective. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). pp. 54–59 (2016)
11. de Oliveira, T.B.F., Bazzan, A.L.C., da Silva, B.C., Grunitzki, R.: Comparing multi-armed bandit algorithms and Q-learning for multiagent action selection: a case study in route choice. In: 2018 International Joint Conference on Neural Networks, IJCNN. pp. 1–8. IEEE, Rio de Janeiro (2018), doi.org/10.1109/IJCNN.2018.8489655
12. Ortúzar, J.d.D., Willumsen, L.G.: Modelling transport. John Wiley & Sons, Chichester, UK, 4 edn. (2011)
13. Rădulescu, R., Mannion, P., Roijers, D., Nowé, A.: Multi-objective multi-agent decision making: a utility-based analysis and survey. Autonomous Agents and Multi-Agent Systems 34 (04 2020)
14. Raith, A., Wang, J.Y., Ehrgott, M., Mitchell, S.A.: Solving multi-objective traffic assignment. Annals of Operations Research 222(1), 483–516 (2014)
15. Ramos, G.de.O., da Silva, B.C., Bazzan, A.L.C.: Learning to minimise regret in route choice. In: Das, S., Durfee, E., Larson, K., Winikoff, M. (eds.) Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017). pp. 846–855. IFAAMAS, São Paulo (May 2017), http://ifaamas.org/Proceedings/aamas2017/pdfs/p846.pdf
16. Santos, G.D.dos., Bazzan, A.L.C.: A multiobjective reinforcement learning approach to trip building. In: Bazzan, A.L., Dusparic, I., Lujak, M., Vizzari, G. (eds.) Proc. of the 12th International Workshop on Agents in Traffic and Transportation (ATT 2022). vol. 3173, pp. 160–174. CEUR-WS.org (2022), http://ceur-ws.org/Vol-3173/11.pdf
17. Santos, G.D.dos., Bazzan, A.L.C., Baumgardt, A.P.: Using car to infrastructure communication to accelerate learning in route choice. Journal of Information and Data Management 12(2) (2021), sol.sbc.org.br/journals/index.php/jidm/article/view/1935
18. Santos, G.D.dos., Bazzan, A.L.C.: Accelerating learning of route choices with C2I: A preliminary investigation. In: Proc. of the VIII Symposium on Knowledge Discovery, Mining and Learning. pp. 41–48. SBC (2020)

19. Santos, G.D.dos., Bazzan, A.L.C.: Sharing diverse information gets driver agents to learn faster: an application in en route trip building. PeerJ Computer Science 7, e428 (March 2021), peerj.com/articles/cs-428/
20. Van Moffaert, K., Nowé, A.: Multi-objective reinforcement learning using sets of Pareto dominating policies. J. Mach. Learn. Res. 15(1), 3483–3512 (Jan 2014)
21. Wang, J.Y.T., Raith, A., Ehrgott, M.: Tolling analysis with bi-objective traffic assignment. In: Ehrgott, M., Naujoks, B., Stewart, T.J., Wallenius, J. (eds.) Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems. pp. 117–129 (2010)
22. Wang, J.Y., Ehrgott, M.: Modelling route choice behaviour in a tolled road network with a time surplus maximisation bi-objective user equilibrium model. Transportation Research Part B: Methodological 57, 342–360 (2013)
23. Watkins, C.: Learning from Delayed Rewards. Ph.D. thesis, University of Cambridge (1989)

Henrique Uhlmann Gobbi is a B.Sc. graduate student in computer science at the Federal University of Rio Grande do Sul, Porto Alegre, Brazil, with an expected graduation date in mid-2025. His research interests primarily revolve around applying artificial intelligence to address real-world problems, in addition to investigating natural science phenomena. He has conducted research in the field of reinforcement learning applied to urban traffic mobility and is currently focusing on the use of artificial neural networks to model problems in neuroscience.

Guilherme Dytz dos Santos is a B.Sc. graduate student in computer science from Federal University of Rio Grande do Sul, Porto Alegre, Brazil, with expected graduation in early 2024. His academic pursuits are centered around research in reinforcement learning, machine learning, and artificial intelligence, all of which are applied to real-world problems in the realm of urban traffic optimization. His research focuses on harnessing the power of reinforcement learning to alleviate the challenges associated with high traffic demands in urban networks.

Ana Bazzan is a professor of Computer Science at the Institute of Informatics at the Universidade Federal do Rio Grande do Sul (UFRGS), where she leads the Artificial Intelligence Group. Her research focuses on multiagent systems, in particular on agent-based modeling and simulation, and multiagent learning. Since 1996, she has collaborated with various researchers in the application of multiagent systems. In recent years, she has contributed to different topics regarding smart cities, focusing on transportation. In 2014, she was General Co-chair of AAMAS (the premier conference in the area of autonomous agents and multiagent systems).

Received: December 10, 2022; Accepted: September 25, 2023.