On improving route choice through learning automata

Gabriel de O. Ramos, Ricardo Grunitzki, and Ana L. C. Bazzan

Instituto de Informática
Universidade Federal do Rio Grande do Sul
Porto Alegre, RS, Brazil
{goramos,rgrunitzki,bazzan}@inf.ufrgs.br

Abstract. Urban mobility is a major challenge in modern societies. Increasing the infrastructure's physical capacity has proven to be unsustainable from a socio-economical perspective. Intelligent transportation systems (ITS) emerge in this context, aiming to make a more efficient use of existing road networks by means of new technologies. In this paper we address the route choice problem, in which drivers need to decide which route to take to reach their destinations. In this respect, we model the problem as a multiagent system where each driver is represented by a learning automaton, and learns to choose routes based on past experiences. In order to improve the learning process, we also propose a mechanism that updates the drivers' set of routes, allowing faster routes to be learned. We show that our approach provides reasonably good solutions, and is able to mitigate congestion levels in main roads.

1 Introduction

The increasing demand for efficient urban mobility is a major challenge in modern societies. Traditional approaches, like increasing the physical capacity, are unsustainable from many perspectives (e.g., economic, environmental). In this context, the concept of intelligent transportation systems (ITS) emerges as a mean to make a more efficient use of existing road networks, trying to mitigate the need for increasing their physical capacity.

One of the fronts towards ITS is concerned with correctly distributing the flow of vehicles into the network to avoid congestions. This kind of problem has been extensively studied by the traffic engineering community [1, 2], and is known as the traffic assignment problem (TAP). The TAP deals with distributing the flow of vehicles in a network regarding its capacity (a.k.a. supply) and the drivers' origins and destinations (OD-pairs, a.k.a. demand).

While traffic assignment is a reasonable approach for traffic engineering, it is not realistic from the drivers' viewpoint. The point here is that, conceptually, traffic assignment is performed by a central authority, which must have complete knowledge about the system as a whole. Although having a central authority is realistic for traffic supervision and infrastructure changes (long term planning), it might be unrealistic for assigning the day-to-day route of each driver (short term planning). The latter case can be referred to as route choice, in which a driver must choose among a set of possible routes from its origin to its destination. Studying the drivers' route choice behaviour is an important consideration to make a more efficient use of road networks [1]. Along these lines, many decentralized approaches have been proposed to the route choice problem, among which multiagent systems have been played a key role [3–5].

In this work, we approach the route choice problem from a multiagent learning perspective. We make use of the learning automata (LA) [6] theory, where a driver-agent (automaton) learns to choose its route based on past experiences. Each agent has a set with the K shortest routes among its origin and destination. The agents learn to choose among these routes using a learning scheme called linear reward-inaction, where routes are chosen with probability inversely proportional to their travel time (i.e., faster routes are more likely to be chosen).

We also propose a simple method to update the drivers' route sets, enabling agents to find alternative routes so as to improve their performance. Based on experiments, we show that our approach is able to find reasonably good solutions, contributing to make a more efficient use of the road network. Furthermore, the impact of our approach in an abstract road network is analysed, concluding that the congestion levels of main roads may be decreased.

Through this work, we aim at contributing towards the adoption of intelligent mechanisms to improve the drivers' decision making, which is aligned with the concept of ITS. From a practical perspective, our approach might be seen as an intelligent mobile service, which can be used by human drivers on their daily route choice process.

This paper is organised as follows. In Sect. 2 we present the related work. The problem is formulated in Sect. 3. Our approach is described in Sect. 4. The experiments and results are discussed and analysed in Sect. 5. Final remarks and future directions are presented in Sect. 6.

2 Related work

Reinforcement learning is used to enable driver-agents to learn their routes in the work of Tavares and Bazzan [3]. To this end, a classical algorithm called Q-learning is used, which represents the environment as a Markov decision process and maps states into actions. In their approach, states are modelled as intersections, and actions as the links that leave the intersections. Through experiments, their approach has shown to be effective. However, as the route learning process is done through trial-and-error, the time required for the route to be learnt may be impractical in real situations. In our approach, we provide a number of alternative routes, among which the agent must learn the better one for him. In this sense, the drivers are able to learn much quicker.

An investigation of how traffic forecasts impact on drivers' decision making is made by Klüegl and Bazzan [4], considering a two-route scenario. In their approach, drivers' route choice is based on a simple heuristic, which consists in selecting a route with probability proportional to its reward. A traffic control system is also proposed, which perceives drivers' decisions and returns a traffic forecast. Based on the forecast, drivers can change their routes before actually driving. The rationale behind the forecasts is to allow drivers to implicitly learn the usual decisions of other drivers. However, the scenario investigated is a quite simple, with just two possible routes. Furthermore, a centralized forecast provider is required.

Bazzan et al. [5] modelled route choice on the basis of the coordination game, where driver-agents deal with the problem of choosing between two routes. In their approach, agents are also able to form groups, and share information about their performance with their peers. In this sense, agents can choose the best route based on their peers' performance. However, such strategy may lead to congestions, since many drivers might choose the same route. Also, an investigation with more routes was left aside.

The route choice problem is addressed with a genetic algorithm by Cagara et al. [7]. In their approach, each driver has k possible routes to choose, which are encoded in a binary way. The individuals' chromosomes represent the routes assigned to each driver. Their approach, however, does not consider the real time experimented on each route, but the travel time under no congestion, which represents a naïve approach. Furthermore, the routes cannot be adapted to improve the drivers' efficiency. In our approach, both issues are appropriately addressed.

Pel and Nicholson [8] proposed a percentile-based route choice model, which assumes that drivers make decisions with respect to the route travel time distributions collected from past experience. In their approach, the focus is on identifying the amount of time that each driver must allocate to its trip in order to reach its destination on time. Considering the inherent variability of the routes travel times according the other drivers' decisions, it was identified that some agents are more sensitive to this variability than others. However, their approach is centralized.

Ortúzar and Willumsen [1] defined two iterative methods for finding approximate solutions for traffic assignment, which serve as a good baseline. The first is the incremental assignment, where the drivers are incrementally assigned into routes, regarding the accumulated vehicles on them. This approach, however, does not allow the traffic to be rearranged so as to improve the final assignment. In this sense, the successive averages method was proposed, which uses an initial assignment based on free flow travel times. Through successive iterations, previous assignments form the basis to update the roads costs, and improve new assignments. However, both approaches are centralized.

The TAP and the road pricing problem (RPP) are jointly addressed by Buriol et al. [2]. Usually, the self-interested behaviour of drivers tends to lead to non-optimal solutions. In this sense, putting tolls in some roads can induce drivers to act in the behalf of a system optimal solution. Thus, RPP was introduced as a mean to achieve such behaviour. In their work, Buriol et al. used a random-key genetic algorithm to find solutions that bridge the gap between the drivers' and the system's optimal solutions. However, their approach depends on a high number of tolls to achieve good results, what may be impractical in economic terms. Furthermore, charging tolls is not a popular alternative from the drivers' viewpoint, although in some situations it may be essential.

As can be seen, many approaches do not consider the impact that agents' decisions and changes in the network have in the final results. Furthermore, many approaches do not provide methods that could be potentially used by drivers without requiring, e.g., a central authority assigning traffic. Our approach, on the other hand, uses an autonomous learning method that is able to consider the stochastic nature of the environment, which is more realistic from the drivers' perspective. Furthermore, our approach could be potentially implemented as an intelligent mobile service, which could be used by drivers in their daily route choice process.

3 Problem formulation

A road network can be modelled as a directed graph $G = \{N, E\}$, where the set N of nodes represents the intersections, and the set E of edges represents the roads between intersections. Each edge $e \in E$ has a cost $c(e)$ associated with crossing it, like travel time or length. In this work, the cost function represents the travel time, and is defined according to Eq. (1) (as in [1]), where f_e and v_e are, respectively, the free flow travel time (minimum travel time when the edge is not congested), and the number of vehicles on the edge $e \in E$.

$$c(e) = f_e + 0.02 \times v_e \tag{1}$$

The flow of vehicles in a network is based on the amount of trips made between different origins and destinations (OD-pairs). A trip is made by means of a route $R \subseteq E$, which is a set of edges that connects an origin to a destination. The cost $C(R)$ of a given route R can obtained by summing up the costs associated with the edges that comprise it, as in Eq. (2).

$$C(R) = \sum_{e \in R} c(e) \tag{2}$$

In this work, the route choice problem is applied in a commuting scenario. A commuter driver repeatedly makes trips under approximately the same conditions (e.g., the daily routine of going to work). In this respect, each commuter is modelled as an agent (driver, henceforth), which repeatedly deals with the problem of choosing routes to make its trips. A driver $i \in D$ aims at choosing the route R that takes the least time to its destination. The utility of driver i when selecting a given route R is associated with the route's cost, as in Eq. (3), i.e., the higher the cost, the lower the utility.

$$u_i(R) = -C(R) \tag{3}$$

4 Method

In this work, the route choice problem is modelled from a LA perspective. A learning automaton (LA) is a reinforcement learning algorithm that attempts to improve agent actions' choice based on past actions, and on the received reinforcement signals. In this work, actions represent routes, and the reinforcement signal received for taking a route represents the utility associated with it. Along these lines, we can formalize a LA by a tuple $\langle \mathcal{R}_i, \pi_i, u_i, \mathcal{L} \rangle$, where:

- $\mathcal{R}_i = \{R_1, ..., R_K\}$ is the set of K routes (actions) available for agent i;
- $\pi_i : R \rightarrow [0,1], R \in \mathcal{R}_i$, is a probability vector over agent i's actions; it is also called policy;
- $u_i : R \rightarrow [0,1], R \in \mathcal{R}_i$, is the utility of agent i after taking route R, i.e., is the reward to be used as reinforcement signal;
- $\mathcal{L}$ is a learning scheme to update agents' π distributions.

A LA learns in episodes. In our approach, an episode is equivalent to a set of trips, one for each driver. Usually, a LA's episode is divided into two parts:

Route choice, where agent i draws a random route $R \in \mathcal{R}_i$ according to its policy $\pi_i(R)$.

Policy update, where the policy π_i is updated using the learning scheme $\mathcal{L}$ based on the received utility (reward) u_i, collected after finishing the episode.

Besides these two steps, in this approach we also employ a third one:

Route set update, where the set of routes of a given agent is recreated based on its experience.

The route choice step is based on the idea that drivers do not always take the route with highest expected utility. In this regard, routes with low utility are also chosen, though less likely than routes with higher utility. This is an important issue because the utility associated with the routes may change in time. In this sense, exploring other routes may bring benefits to the learning process.

The policy update step is performed after each episode based on the received utility, which is associated with the route taken. The update is made by means of a learning scheme $\mathcal{L}$. In this work, we use the well-known linear reward-inaction (L_{R-I}) scheme [6]. The L_{R-I} scheme updates the probability vector π_i according to Eq. (4), where $R^* \in \mathcal{R}_i$ is the route taken by agent i in the current episode, and $\alpha \in [0,1]$ is the learning rate.

$$\pi_i(R) \leftarrow \pi_i(R) + \begin{cases} \alpha u_i(R)(1 - \pi_i(R)) & \text{if } R = R^* \\ -\alpha u_i(R)\pi_i(R) & \text{for the other routes} \end{cases} \tag{4}$$

In order to create the set of routes $\mathcal{R}_i$ of driver i, a classical algorithm called K Shortest Loopless Paths (KSP) [9] is used, which is able to find the K shortest routes between an OD-pair. Roughly speaking, the KSP algorithm starts with the shortest route, iteratively removing each edge from it and calculating a new route, until K shortest routes are found. The sets of routes are generated while creating the agents. At this time, the routes are generated considering free flow travel time[1] on edges, i.e., Eq. (1) with low values for v_e that still allow free flow travel time. Such an approach represents a suitable initial approximation for guiding the route choice policy. As the agents become experienced, however, the policy update step undertakes the task of incorporating the real edges' utility (reward) into the policy.

The policy update step improves on the route choice process by increasing the importance of routes that improve the agents' utilities. However, as the agents become experienced, alternative routes may prove appealing to them. Also, the utility of routes that were initially good may start to deteriorate with the increased flow of vehicles. In this sense, we also employ a route set update step, which is performed soon after the policy update step. To this end, each driver internally stores the mean costs experimented on each edge travelled, as shown in Eq. (5), where $\overline{c}_i(e)$ is the mean travel time experimented by agent i on edge e. Through this information, with small probability ω, each driver is able to rerun the KSP algorithm regarding their own estimations about edges travel times, and obtaining a new set of routes. Such an approach is used because, through experience obtained from previous episodes, drivers are able to find better routes. Furthermore, this mechanism promotes a higher diversity of routes (better distributing the flow of vehicles within the road network).

$$S_i = \{\langle e, \overline{c}_i(e)\rangle : e \in E\} \tag{5}$$

5 Experiments and results

In order to evaluate the performance of our LA-based approach, we use the road network proposed by Ortúzar and Willumsen in the exercise 10.1 of their book [1] (OW10.1, henceforth). The OW10.1 network is composed by 13 nodes (named $A, B, ..., M$) and 24 bi-directed edges, as shown in Fig. 1, where the edges' numbers represent their free flow travel time (in minutes). The referred network has four OD-pairs: AL, AM, BL, and BM, with demand of 600, 400, 300, and 400 vehicles, respectively. Experiments were conducted through macroscopic modelling and simulations.

The OW10.1 network was chosen because it is small and has many problems related to routes overlapping. Table 1 presents, for each OD-pair, the shortest routes under no congestion (R^- hereafter), the cost under no congestion, and the cost when all drivers take the same routes R^- (regarding their OD-pairs), which will be considered as a baseline in our experiments. As can be seen, if the route AL (second column of Table 1) were used by all the 600 drivers of AL pair, then its cost would increase from 28 to 115 minutes. Another important observation is that the BM route is much faster than others. This happens because the other

[1] We assume that drivers are aware of the network topology (and also free flow travel times). This is a realistic assumption since this kind of information is already provided by navigation devices (such as GPS). On the other hand, drivers do not know in advance the traffic conditions on the network, what influence the real travel time on each edge.

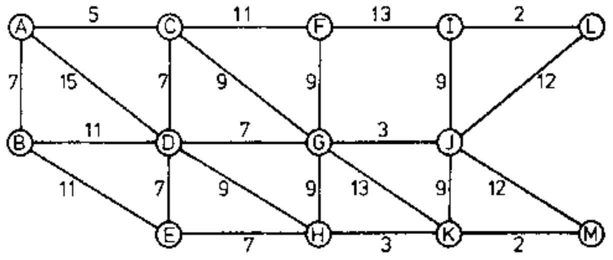


Fig. 1: OW10.1 road network (extracted from [1])

pairs' routes are overlapping at least in some point. For example, in the second hop, node G receives flow from both pairs AL and BL pairs. At that specific point, 900 vehicles are trying to use the same edge GJ. In this respect, the OW10.1 network seems an interesting validation scenario.

In order to better evaluate our approach, we measure the drivers' performance by means of the relative improvement achieved by them. The relative improvement measures how much a route R_i improves driver i's utility over route R_i^- (i.e., against last column of Table 1). In this sense, the overall performance of the traffic network may be obtained through the mean relative improvement of drivers, as in Eq. (6). The ratio $\frac{C(R_i)}{C(R_i^-)}$ denotes the relative improvement obtained by driver i, where values lower than 1 mean that the utility has been improved, and values greater than 1 mean that the driver's utility has deteriorated.

$$\sqrt[|D|]{\prod_{i=1}^{|D|} \frac{C(R_i)}{C(R_i^-)}} \tag{6}$$

In all experiments, the number of episodes was empirically set to 500. The values set to parameters K, ω, and α are discussed in Sect. 5.1. A comparison of our approach against others is presented in Sect. 5.2. Aspects concerning the learning convergence are discussed in Sect. 5.3. Finally, a brief study about the impact of our approach in the OW10.1 network is presented in Sect. 5.4.

Table 1: Shortest routes (under no congestion) for each OD-pair, and respective costs under no congestion and when all drivers take the same route

OD-pair	Shortest route under no congestion	Route cost (min)	
		Under no congestion	When all OD-pairs' drivers take the same route
AL	A - C - G - J - I - L	28	115
AM	A - C - D - H - K - M	26	79
BL	B - D - G - J - I - L	32	98
BM	B - E - H - K - M	23	55

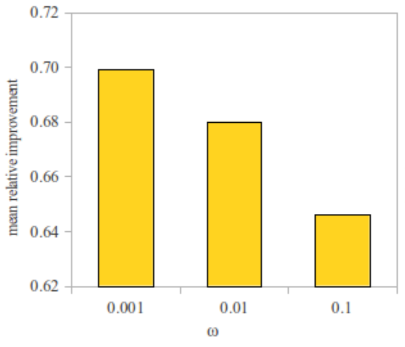
5.1 Parameters

In this first set of experiments, we analyse the impact of the parameters K, α and ω in the systems' performance. First of all, we emphasize that results and conclusions presented here are based exclusively on the OW10.1 network, and do not necessarily hold in other networks without a further investigation.

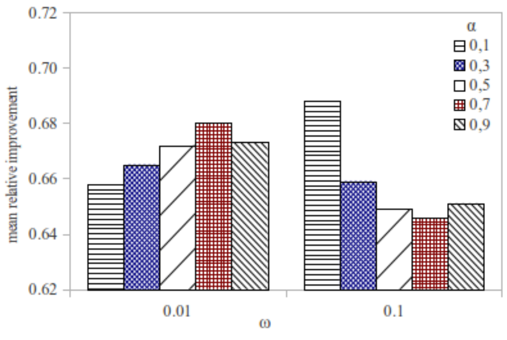
We begin by trying different values for K and α, and deactivating our route set update step ($\omega = 0$). After experimenting different values for K, its impact on the result has shown inexpressive. However, based on experiments, we realised that $K = 8$ may be the best choice for the OW10.1 network, improving the solution by up to 3% when compared to other values (we omit the plot due to the lack of space). Concerning parameter α, we concluded that its value does not impact directly in the final result when $\omega = 0$. However, when the route set update step is activated ($\omega > 0$), the value of the parameter starts to also impact the result (as shown forward).

The routes set update step used in our approach is responsible for enhancing the diversity of routes. With more routes being used, a more efficient usage of the network is expected. Figure 2a presents the impact of different values of ω in the results, with $K = 8$ and $\alpha = 0.7$. As shown, when routes sets are updated more frequently, the final result is improved. That is because higher values for ω increase the routes' diversity, i.e., more edges are explored. Consequently, the flow of vehicles is better distributed across the network.

When the step of updating the routes set is performed, the value of parameter α has to be considered. In this sense, different values of α were tested against different values of ω so as to find a suitable combination of values for these parameters. Figure 2b presents these results, with $K = 8$. As can be seen, increasing α when $\omega = 0.01$ deteriorates the results. In turn, when $\omega = 0.1$, increasing α leads to improving the results. The rationale behind this phenomenon is that, when



(a) ω



(b) ω vs. α

Fig. 2: Mean relative improvement for different values of: ω, with $K = 8$ and $\alpha = 0.7$ (Fig. 2a); and ω vs. α, with $K = 8$ (Fig. 2b)

ω is higher, the set of routes of agents changes faster, and new routes may be learned from scratch. As a consequence, a higher α (up to certain limits) makes agents learn faster. The benefit from such a situation is that more alternative routes are explored, thus agents are more likely to improve their utility. When $\omega = 0.01$, however, the scenario is inverse: with fewer alternative routes being used, high values for α pose disturbances in the learning process, deteriorating the agents' utility.

Therefore, the best combination of parameters for this specific network is $K = 8$, $\alpha = 0.7$, and $\omega = 0.1$. These values are used in the next subsections' experiments.

5.2 Comparison against other methods

In this section we compare our LA-based approach with other methods from the literature. The focus here is on comparing the efficiency of the approaches on generating good solutions, from the perspective of the mean relative improvement measure. The methods used in the comparison are:

- Incremental and successive averages [1], two classical traffic engineering approaches for the traffic assignment problem.
- Q-learning for learning routes on the fly (similar to [3]). The Q-learning parameters were empirically set as $\alpha = 0.05$, $\gamma = 0.99$, and $\epsilon = 1.0$ coupled with a decay rate of 0.995, which is multiplied by ϵ at the end of each episode.

The results are presented in Table 2. As can be seen, our approach outperforms the others. Compared to the traffic assignment methods described in [1], our approach has the advantage of being decentralized (i.e., it does not require a central authority), which is more realistic from the route choice perspective.

Compared to Q-learning, the result of our approach is slightly better. The Q-learning approach has shown effective for learning routes on the fly. However, from a practical perspective, our approach is more suitable for real situations than Q-learning, since it learns faster (as shown in next section).

5.3 Convergence and learning speed

In this section we compare our LA-based approach against the Q-learning one regarding convergence and learning speed. A comparison in terms of convergence

Table 2: Relative improvement of our LA-based approach and other methods

Method	AM	AL	BL	BM	All
LA-based	0.502±0.03	0.664±0.04	0.582±0.03	0.896±0.07	0.647±0.16
Q-learning	0.52±0.07	0.704±0.06	0.606±0.04	0.934±0.11	0.676±0.18
Incremental	0.589	0.837	0.708	1.145	0.772
Successive averages	0.614	0.82	0.703	1.131	0.777

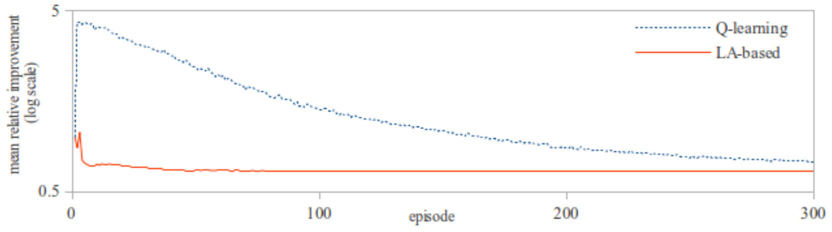


Fig. 3: Convergence time of our LA-based approach against Q-learning

is presented in Fig. 3 (we show up to 300 episodes so as to improve visualisation). The comparison takes place through the mean relative improvement along episodes. As shown, convergence was reached by our approach in less than 50 episodes. This occurs because the policy update scheme rapidly and efficiently learns the expected utility of each route. On the other hand, the complexity of the Q-learning approach increases with the number of possible states and actions. As a consequence, the Q-learning approach requires about to 300 episodes to find a good policy.

In terms of learning speed, our approach is also faster. The reason behind this behaviour relates to the fact that the Q-learning approach needs to learn the routes on the fly. In other words, all possible combinations of actions and states might be tested before a good solution is reached. The routes are improved as the model of the environment is learnt. On the other hand, our approach has an initial set of possible routes. This set of routes can be improved based on the drivers' experience (according to parameter ω). Furthermore, the dynamics of the environment are indirectly embedded in the probability vector over the set of routes, which is used to guide the route choice.

5.4 Impact in the network

In order to evaluate the impact of our approach in the OW10.1 network, we show how the edges occupancy varies along episodes. The occupancy rate of an edge is given by dividing the number of vehicles on it by its capacity[2]. When the occupancy rate of an edge nears 0, then there are few vehicles on it; on the other side, when an edge is under congestion, this rate nears 1. It should be noted that the occupancy rate may be larger than one. This is due to the fact that, in macroscopic simulations, the number of vehicles on a given edge is not constrained to its capacity (as observed in the literature [1]). However, albeit one may say that this measure is quite artificial, it is still a suitable approach to evaluate the congestions levels in the edges.

[2] The capacity of an edge is a function of its length. We estimate the length of an edge by assuming a maximum speed of 50km/h, and multiplying it by the edge's travel time under no congestion. In this sense, the capacity of a route may be approximated by dividing its estimated length by 5 (given that, in literature, 5m is a good approximation for the vehicles' lengths and the space between them).

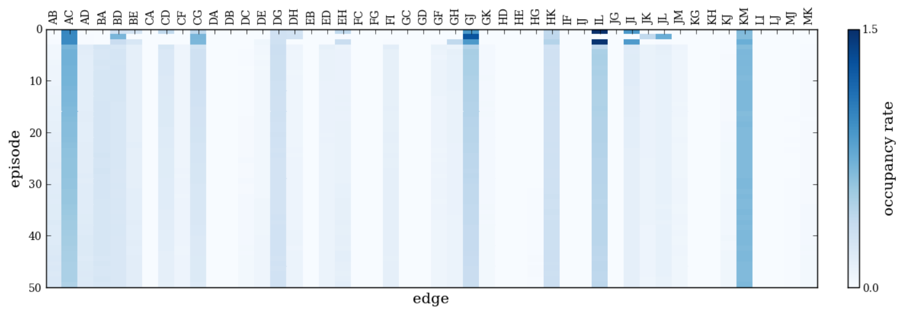


Fig. 4: Edges' occupancy rate along episodes

Figure 4 presents the edges' occupancy rate along episodes by means of a heat-map. In the figure, horizontal axis represents the edges, and vertical axis represents the episodes. For each edge and episode, the plot shows the highest occupancy rate measured in the edge in that specific episode. In the plot, the darker the colour, the higher the occupancy rate. Results are shown for the 50 first episodes (where higher variations occur).

As shown in the plot, the occupancy rate of some edges is not improved along time, as is the case of edge KM. However, most of the edges have improved their occupancy rate. For instance, the initial occupancy rate of edge AC was of almost 1 (high congestion). However, after our approach was used, such a rate dropped to 0.35. This improvement confirms our initial hypothesis that, as drivers learn to choose their routes, the use of the road network becomes more efficient.

Another important observation regards the fact that some edges have worsen their occupancy rate (e.g., edge AD). That is a natural effect of distributing the drivers into the road network. In the initial scenario (all drivers taking the same routes), just a few edges absorb all the network flow, and all the unused edges have a zero occupancy rate. However, as the traffic is distributed across the network, part of the flow of vehicles is diverted from the most used edges into those that were not being used before. Therefore, it is clear that the aim here is not on reducing the occupancy rate of all edges, but distributing the flow in a way that the drivers' utility, and also the overall network utility, is maximized.

6 Conclusions

In this paper, we have presented a multiagent approach to address the route choice problem. The problem is approached from a LA perspective, where driver-agents learn to choose routes based on past experience. The set of routes is given in advance to the agents, and their learning scheme is based on the linear reward-inaction algorithm. A simple route set update is proposed, which promotes routes' diversity. Through experiments, we have shown that our approach provides reasonably good solutions, and is able to make a more efficient use of the road network. A key point of our approach related to traditional ones, is that ours does not require a central authority assigning traffic. Instead, we make use of learning techniques that allow the traffic elements (drivers, in this case) to improve their, and system's, efficiency. Furthermore, our approach might be potentially implemented as an intelligent mobile service to guide the drivers' daily route choice process.

For future work, we would like to investigate information exchange mechanisms, like inter-vehicular communication (IVC). Through IVC, agents can improve their knowledge about network conditions, and also can benefit from social interactions. Another interesting direction would be enhancing the route set update mechanism. An alternative in this direction would be replacing the probability selection (with ω) by a domain aware criterion (e.g., if the performance of a driver deteriorates below a given rate). Modelling the network performance in the agents' utility also represents a promising direction. Finally, other algorithms for learning (instead of L_{R-I}) and routes generation (instead of KSP) may be a good step towards improving our approach.

7 Acknowledgments

Ana L. C. Bazzan and Gabriel de Oliveira Ramos are partially supported by CNPq and FAPERGS in their works (grants 305062 and 10/0049-7).

References

1. Ortúzar, J., Willumsen, L.: Modelling Transport. 3rd ed. John Wiley & Sons (2001)
2. Buriol, L.S., Hirsh, M.J., Pardalos, P.M., Querido, T., Resende, M.G., Ritt, M.: A biased random-key genetic algorithm for road congestion minimization. Optimization Letters **4** (2010) 619–633
3. Tavares, A.R., Bazzan, A.L.C.: Independent learners in abstract traffic scenarios. Revista de Informática Teórica e Aplicada **19**(2) (2012) 13–33
4. Klügl, F., Bazzan, A.L.C.: Route decision behaviour in a commuting scenario. Journal of Artificial Societies and Social Simulation **7**(1) (2004)
5. Bazzan, A.L.C., Fehler, M., Klügl, F.: Learning to coordinate in a network of social drivers: The role of information. In Tuyls, K., Hoen, P.J., Verbeeck, K., Sen, S., eds.: Proc. of the Intl. Workshop on Learning and Adaptation in MAS (LAMAS 2005). Number 3898 in Lecture Notes in Artificial Intelligence (2006) 115–128
6. Narendra, K.S., Thathachar, M.A.L.: Learning Automata: An Introduction. Prentice-Hall, Upper Saddle River, NJ, USA (1989)
7. Cagara, D., Scheuermann, B., Bazzan, A.L.C.: A Methodology to Evaluate the Optimization Potential of Co-ordinated Vehicular Route Choices. In: 1st GI/ITG KuVS Fachgespräch Inter-Vehicle Communication (FG-IVC 2013), Innsbruck (2013)
8. Pel, A.J., Nicholson, A.J.: Network effects of percentile-based route choice behaviour for stochastic travel times under exogenous capacity variations. In: Proc. of the 16th Intl. IEEE Annual Conf. on Intelligent Transportation Systems. (2013) 1864–1869
9. Yen, J.Y.: Finding the k shortest loopless paths in a network. Management Science **17**(11) (1971) 712–716