On Estimating Action Regret and Learning From It in Route Choice

Gabriel de O. Ramos and Ana L. C. Bazzan
Instituto de Informática
Universidade Federal do Rio Grande do Sul
{goramos,bazzan}@inf.ufrgs.br

Abstract

The notion of regret has been extensively employed to measure the performance of reinforcement learning agents. The regret of an agent measures how much worse it performs following its current policy in comparison to following the best possible policy. As such, measuring regret requires complete knowledge of the environment. However, such an assumption is not realistic in most multiagent scenarios. In this paper, we address the route choice problem, in which each driver must choose the best route between its origin and its destination. The expected outcome corresponds to an equilibrium point in the space of policies where no driver benefits from deviating from its policy, a concept known as User Equilibrium (UE). Considering the limited observability of such a scenario, we investigate how the agents can estimate their regret based exclusively on their experience. To this regard, we introduce the concept of estimated action regret, through which an agent can estimate how much worsen it performs by taking a given action rather than the best in hindsight. Additionally, we show how such estimations can be used as a reinforcement signal to improve their performance. We empirically evaluate our approach in different route choice scenarios, showing that the agents produce reasonable estimations of their regret. Furthermore, we show that using such estimations as the reinforcement signal provides good approximations to the UE.

1 Introduction

Reinforcement learning (RL) in multiagent domains is a challenging task. In RL, an agent must learn by trial-and-error how to behave within the environment in order to maximise its utility. When multiple agents share a common environment, they must adapt their behaviour to those of others. The problem becomes even harder when the agents are selfish and compete for a common resource. An example is the route choice problem, which concerns how rational drivers[1] behave when choosing routes between their origins and destinations to minimise their travel costs. Learning is a fundamental aspect of route choice because the agents must adapt their choices to account for the changing traffic conditions. In other words, the agents must adapt to each others' decisions.

An interesting class of multiagent RL techniques comprises the regret minimisation approaches. In this context, regret has been typically employed to measure the performance of reinforcement learning agents. Specifically, regret measures how much worse an agent performs following its current policy in comparison to following the best possible policy one in hindsight. In this sense, regret minimisation can be seen as an inherent definition on how rational agents behave over time. Along these lines, the regret measure naturally fits as a guide of the learning process.

In recent works [Zinkevich *et al.*, 2008; Bowling and Zinkevich, 2012; Waugh *et al.*, 2015], regret has been used to improve the learning process. However, calculating regret requires complete knowledge of the environment (i.e., the utility associated with every possible policy). In fact, one may assume that an online service broadcasts the required information through mobile devices. Nevertheless, investigating methods to accomplish such a task in the absence of any global information is more challenging and is also relevant, especially in highly competitive scenarios like traffic [Bazzan and Klügl, 2013; Stone and Veloso, 2000].

In this paper, we address the route choice problem by minimising regret. Specifically, we investigate how the agents can estimate their regret locally (i.e., based exclusively on their experience) and how such estimations can be employed to guide the RL process. To this regard, each agent keeps an internal history of experienced rewards, which is used for estimating the regret associated with each of its actions. We refer to such measure as the *estimated action regret* and employ it for updating the agents' policies. The expected outcome corresponds to an equilibrium point in the space of policies in which no driver benefits from deviating from its policy. This is the so-called User Equilibrium (UE) [Wardrop, 1952]. To the best of our knowledge, this is the first attempt to improve the learning process by using regret estimations as the reinforcement signal.

Through experiments, we show that our approach provides fairly precise estimations of the agents' regret relying only on agents' experience. Moreover, we present good evidence that using such regret estimates as the reinforcement signal is

[1] Henceforth, we use the terms agent and driver alternately.

beneficial for the learning process. Consequently, in all tested cases, the results are reasonably close to the UE.

We remark that this work represents our very first step towards developing rational agents able to analyse their learning performance and to improve their expected outcome. In the medium-term, we aim at investigating formal aspects of the learning process to guarantee the efficiency of RL under multiagent domains.

This paper is organised as follows. The background on route choice, RL and regret algorithms is presented in Section 2. In Sections 3 and 4, we describe how the agents can estimate their regret locally and how they can learn using such estimations, respectively. The experimental evaluation is discussed in Section 5. Finally, Section 6 presents the concluding remarks and future work directions.

2 Background

2.1 Route Choice

The route choice problem concerns how rational drivers behave when choosing routes between their origins and destinations. In this section, we introduce the basic concepts related to route choice. For a more comprehensive overview, we refer the reader to [Ortúzar and Willumsen, 2011].

A road network can be represented as a directed graph $G = (N, L)$, where the set of nodes N represent the intersections and the set of links L represent the roads between intersections. The demand for trips generates a flow of vehicles on the links, with f_l the flow on link l. To this regard, each link $l \in L$ has a cost $c_l : f_l \to \mathbb{R}^+$ associated with crossing it. Let T_{ij} be the demand for trips between origin $i \in N$ and destination $j \in N$ (we refer to an origin-destination pair as simply OD pair). The set of all such demands is represented by an OD matrix $T = \{T_{ij} \mid \forall i, j \in N,\ i \neq j,\ T_{ij} \geq 0\}$. The total demand is denoted $d = \sum_{T_{ij} \in T} T_{ij}$. A trip is made by means of a route $R \subseteq L$, which is a sequence of links connecting an origin to a destination. The cost of a route R is given by $C_R = \sum_{l \in R} c_l$.

The cost of a link is typically modelled using the volume-delay function (VDF) abstraction. A VDF takes as input the flow of vehicles within a link and, based on its attributes (such as length and capacity), returns the cost (travel time) of such link. A simple VDF is presented in Equation (1), with t_l denoting the free flow travel time (i.e., minimum travel time, when the link is not congested). In this particular VDF, the travel time on link l is increased by 0.02 for each vehicle/hour of flow.

$$c_l(f_l) = t_l + 0.02 \times f_l \tag{1}$$

In the route choice process, drivers decide which route to take every day to reach their destinations. Usually, this process is modelled as a commuting scenario, where drivers' daily trips occur under approximately the same conditions. In this sense, each driver $i \in D$, with $|D| = d$, is modelled as an agent, which repeatedly deals with the problem of choosing the route that takes the least time to its destination. The utility $u_i : R \to \mathbb{R}$ received by driver i after taking route R is inversely associated with the route's cost, as in Equation (2). The expected outcome of this problem can be described by the Wardrop's first principle: "the cost on all the routes actually used are equal, and less than those which would be experienced by a single vehicle on any unused route" [Wardrop, 1952]. Such a solution concept is known as User Equilibrium (UE). However, observe that the UE does not describe the system at its best operation. Indeed, such an state is only achieved when the average travel cost is minimum, as described by the Wardrop's second principle [Wardrop, 1952]. To this regard, such solution concept is commonly referred as System Optimum (SO).

$$u_i(R) = -C_R \tag{2}$$

2.2 Reinforcement Learning

Reinforcement learning (RL) concerns with how an agent learns a behaviour by reward and punishment from interactions with its environment. We can formulate the RL problem as a Markov decision process (MDP). An MDP is a tuple $\langle S, A, T, r \rangle$, where: S is the set of environment states; A is the set of actions; $T : S \times A \times S \to [0, 1]$ is the state transition function, with $T(s, a, s') = P(s' \mid s, a)$ representing the probability of reaching state s' after taking action a in state s; and $r : S \times A \to \mathbb{R}$ is the reward function, with $r(s, a)$ denoting the reward received after taking action a in state s [Sutton and Barto, 1998].

In the context of the route choice problem, the actions of an agent represent the choice of routes between his origin and destination. Such a set of actions is known a priori by the agents. In this sense, the set of states and, consequently, the transition functions can be ignored. Moreover, we can define the reward received after taking action a as $r(a) = u(R)$, with $a = R$ (we will refer to reward and utility, rather than cost, hereinafter). On this basis, we can model the route choice problem as a stateless[2] MDP.

Solving a stateless MDP involves finding a policy π (i.e., which route to take) that maximises the accumulated reward. When the model of the environment dynamics (i.e., the reward function r) are known by the agent, finding such an optimal policy is trivial. However, this is rarely the case, especially in multiagent settings. To this regard, the agent must repeatedly interact with the environment to learn a model of its dynamics. A particularly suitable class of RL algorithms here comprises the so-called temporal-difference algorithms, through which an agent can learn without an explicit model of the environment.

The Q-learning algorithm is a good representative of temporal-difference methods [Watkins and Dayan, 1992]. In the case of a stateless MDP, a Q-learning agent learns the expected return $Q(a)$ for selecting each action a by exploring the environment. The exploration of the environment must balance exploration (gain of knowledge) and exploitation (use of knowledge). A typical strategy here is ϵ-greedy, in which the agent chooses a random action with probability ϵ (exploration) or choosing the best action with probability $1 - \epsilon$ (ex-

[2]Observe that a stateless MDP is equivalent to having an initial state with actions corresponding to the routes available to the agent, and an ending state with no actions. When an agent chooses action a on the initial state, it performs the desired action and reaches the ending state with probability 1, receiving reward $r(a)$.

ploitation). Usually, ϵ starts with 1.0 and, at the end of each learning episode, it is multiplied by a decay rate λ in order to increase exploitation as the agent converges to its best action. After taking action a and receiving reward $r(a)$, the stateless Q-learning algorithm updates $Q(a)$ using Equation (3), where the learning rate $\alpha \in [0, 1]$ weights how much of the previous estimate should be retained. The Q-learning algorithm is guaranteed to converge to an optimal policy in the limit under certain assumptions.

$$Q(a) = (1 - \alpha)Q(a) + \alpha r(a) \tag{3}$$

2.3 Regret

The regret concept was introduced in the context of evaluating the performance of learning rules [Hannan, 1957]. Regret measures how much worse an agent i performs, on average, by following its current policy $\pi_i \in \Pi$ as compared to following the best possible policy in hindsight. Precisely, the regret $\mathcal{R}_i^T$ of agent i at time T is given by Equation (4), where $r^t(a)$ represents the reward of action a at time t and, slightly abusing notation, π^t represents the action taken at time t under policy π. Therefore, regret can be seen as the average amount lost for following a policy other than the best one.

$$\mathcal{R}_i^T = \max_{\pi \in \Pi} \frac{1}{T} \sum_{t=1}^{T} r^t(\pi^t) - \frac{1}{T} \sum_{t=1}^{T} r^t(\pi_i^t) \tag{4}$$

In the context of reinforcement learning (RL), regret has been typically used as a measure of convergence [Shoham *et al.*, 2007; Buşoniu *et al.*, 2008; Zinkevich, 2003; Bowling and Veloso, 2002; Powers and Shoham, 2005; Banerjee and Peng, 2005]. An RL algorithm is said no-regret if it learns a policy π for which $\mathcal{R}^T \to 0$ as $T \to \infty$ [Hannan, 1957]. Along these lines, regret minimisation can be seen as a natural definition on how rational agents behave over time. In this paper, we use the regret measure to guide the learning process.

We remark that, by definition, computing regret assumes complete knowledge of the environment. Specifically, an agent cannot compute its regret without knowing the reward of every other action along time. Consequently, agents cannot (i) calculate their regret and (ii) learn using their regret, except if very strong assumptions are made (e.g., assuming that every agent knows the reward of all actions along time). Therefore, using the regret of Equation (4) to guide the learning process is not realistic in multiagent scenarios.

Zinkevich et al. introduced the concept of counterfactual regret and proposed an algorithm for minimising it [Zinkevich *et al.*, 2008]. The counterfactual regret is used to estimate the regret when the information about states is incomplete (useful in extensive form games with imperfect information). This is one of the first works to include the regret in the learning process. Subsequently, Waugh et al. employed a regression algorithm to provide enhanced estimates on the counterfactual regret [Waugh *et al.*, 2015]. However, these works assume that the regret is known by the agents, which is unrealistic for the problem we are considering.

In [Bowling and Zinkevich, 2012], the authors propose a graph representation to express the relation between actions and the associated regret. Such a representation was employed to mimic the structure of local search methods, thus allowing no-regret algorithms to minimise a broader class of optimisation problems. Nevertheless, their work also assumes that the utility function is available to the agents.

Powers and Shoham proposed a set of performance criteria regarding multiagent learning and proposed an algorithm that meets such criteria [Powers and Shoham, 2005]. Their algorithm, however, makes some strong assumptions regarding the environment observability (e.g., an agent can observe its opponents' rewards). Banerjee and Peng extended Powers and Shoham's approach to a class of no-regret algorithms and dropped some of the observability assumptions [Banerjee and Peng, 2005]. Notwithstanding, these approaches do not employ the regret to guide the learning process.

Along these lines, in this paper we investigate how agents can *estimate* their regret based on their experience and propose the use of such estimations to guide the agents' learning process. Moreover, we disaggregate the regret formulation by measuring the regret of actions rather than of policies. We show that performing such estimates is realistic and improves the learning process. To the best of our knowledge, this is the first effort towards addressing regret estimation and learning through such regret.

3 Estimating Regret Locally

In this section, we discuss how agents can estimate their regret. As discussed in Section 2.3, the agents cannot compute their real regret (using Equation (4)) due to the lack of information regarding the routes rewards. The point is that regret is measured considering (i) the agent's average reward under their current policy and (ii) the average reward under the best policy in hindsight. Calculating the latter requires knowing the rewards of all routes along time. However, after each trip, an agent can observe the reward of the route taken, but cannot observe the reward of the other routes. Such a full observability property would only be possible under strong assumptions (e.g., a central authority broadcasting such information), which can be unrealistic in traffic domains. Furthermore, investigating methods to accomplish such a task in the absence of any supporting service is more challenging and is also relevant, especially in the highly competitive settings considered here [Stone and Veloso, 2000].

In this paper, we investigate how agents can estimate their regret based exclusively on local information (i.e., the rewards actually experienced by them). To this regard, we propose an alternative definition of regret that describes the estimated regret of each action.

Let $A_i \subseteq A$ denote[3] the set of routes of agent i. At time t, agent i performs a given action $a_i^t \in A_i$ and receives a reward of $r^t(a_i^t)$. We represent the history of experiences of agent i as $H_i = \{r_{i,a}^t \mid a \in A_i, t \in [1, T]\}$, with $r_{i,a}^t$ the reward experience of driver i for taking action a at time t. However, recall that an agent cannot observe the reward of action a on time t except if it has taken such action at that time, i.e., if $a = a_i^t$. To this regard, the agents can assume the reward of non taken actions to be the same as the most up to date experience on that route. Precisely, let $\tilde{r}_{i,a}^t$ represent the newest reward

[3] We slightly abuse notation here to account for drivers with different OD pairs, whose route sets are obviously different.

experience of agent i for taking action a on time t (either the current reward or the last[4] actually experienced one), as given by Equation (5). The history of experiences of agent i can then be rewritten as $H_i = \{\tilde{r}^t_{i,a} \mid a \in A_i, t \in [1,T]\}$.

$$\tilde{r}^t_{i,a} = \begin{cases} r^t(a^t_i) & \text{if } a = a^t_i \\ \tilde{r}^{t-1}_{i,a} & \text{otherwise} \end{cases} \quad (5)$$

Given the above definitions, we can now formulate the average estimated regret of agent i for taking action a at time t as in Equation (6). In general terms, we will refer to this formulation as *estimated action regret*. The estimated action regret $\tilde{\mathcal{R}}^t_{i,a}$ can be seen as the estimated amount lost by agent i for taking action a at time t instead of the best action regarding its experience. Additionally, we can reformulate Equation (4) to obtain the *estimated agent regret*, as in Equation (7). The estimated agent regret $\tilde{\mathcal{R}}^t_i$ expresses how much worse agent i performed, on average, up to time t for not taking only the best action regarding its experience. The main advantage of this formulation over the real regret (Equation (4)) is that it can be computed locally by the agents, eliminating the need for a central authority. Moreover, as the required information is already available to the agents, they can use such regret to guide their learning process.

$$\tilde{\mathcal{R}}^t_{i,a} = \max_{b \in A_i} \frac{1}{t} \sum_{s=1}^{t} \tilde{r}^s_{i,b} - \frac{1}{t} \sum_{s=1}^{t} \tilde{r}^s_{i,a} \quad (6)$$

$$\tilde{\mathcal{R}}^t_i = \max_{a \in A_i} \frac{1}{t} \sum_{s=1}^{t} \tilde{r}^s_{i,a} - \frac{1}{t} \sum_{s=1}^{s} r^s(a^s_i) \quad (7)$$

4 Learning Through Regret

In this section, we present a simple algorithmic solution for the agents to learn using the estimated action regret of Equation (6). To this end, we employ the Q-learning algorithm (as presented in Section 2.1). We highlight, however, that any other reinforcement learning algorithm could be used as well.

Every driver $i \in D$ is represented by a Q-learning agent. The route choice problem can then be modelled as a stateless MDP. As such, the states and the transition functions can be ignored. Let $A = \{A_i \mid i \in D\}$ be the set of agents' actions, where A_i is the set of routes of agent i, with $A_i = A_j$ if agents i and j belong to the same OD pair. The reward for taking action a at time t is given by $r^t(a)$.

The learning process works as follows. At each episode $t \in [1,T]$, each agent $i \in D$ chooses an action $a^t_i \in A_i$ using the ϵ-greedy strategy. After taking the chosen action, the agent receives a reward of $r^t(a^t_i)$. Afterwards, the agent updates its history H_i using Equation (5) and calculates the estimated regret of action a^t_i using Equation (6). Finally, the agent updates $Q_i(a^t_i)$ using Equation (3). This process is repeated for each episode, aiming at minimising agents' regret.

Recall that the original definition of regret of Equation (4) measures the regret of the agent, not of his actions. However, the agent regret is not useful in the learning process because it does not consider how much the reward of a single action contributes to the regret. In other words, as we consider the average regret, the reward of a good-performing action could be penalised by those of bad-performing actions. Moreover, the learning process works by adjusting the expected value (or regret) of each action of the agent. In this sense, our estimated action regret definition isolates the regret per action, thus allowing the actions to be evaluated singly. The estimated action regret is more suitable to evaluate how promising a given action is as compared to the others.

Finally, it is worth noting that, although each driver minimises its actions' regret, this is equivalent to minimising its total regret. Recall that the estimated action regret measures how much worse an action performs as compared to the best one. By employing such a value in the learning process, the agent puts more weight on the actions with smaller regret. Moreover, using the ϵ-greedy strategy, the agent tends to choose the action with the smallest regret with a higher probability. Consequently, we can state that minimising the estimated action regret along time is equivalent to minimising the estimated agent regret.

5 Experimental Evaluation

5.1 Setup

In order to evaluate our approach, we employ five different road networks that are available in the literature.

`Pigou` : this is a classical network introduced in [Pigou, 1920]. It comprises only two links l_1 and l_2, with $c_{l_1}(f_{l_1}) = 1.0$ and $c_{l_2}(f_{l_2}) = f_{l_2}/d$. We set the number of drivers to $d = 100$, all of them belonging to the same OD pair. In this scenario, there are only two actions (one corresponding to each link), i.e., $|A| = 2$.

`OW` : is a small, synthetic network, with $|N| = 13$, $|L| = 48$, and $d = 1700$ [Ortúzar and Willumsen, 2011, exercise 10.1]. The vehicles are distributed among four OD pairs. The costs of the links are calculated using Equation (1). In this network, the number of possible routes for each OD pair is high. To this regard, we employ the KSP algorithm [Yen, 1971] to find the k shortest routes for each OD pair, i.e., $|A| = k$.

`Braess graphs` : these are expanded versions of the network introduced in [Braess, 1968]. Specifically, let $p \in \{1, 2, \ldots\}$ be the p^{th} expansion of such graph, where `1st Braess graph` is equivalent to the original graph [Roughgarden, 2006]. We generalise such model to consider a discrete set of drivers. The complete description of these graphs is available in Appendix A. We employ the `1st Braess graph`, `2nd Braess graph` and `3rd Braess graph`, and define $d = 4000$, with all drivers belonging to the same OD pair, and, by definition, $|A| = 2p + 1$.

An experiment corresponds to a complete execution, with 1000 episodes, of the Q-learning algorithm on a single network. After an execution is completed, we measure its performance by means of the average travel time (*avg-tt* hereafter, measured in minutes) and the average estimated agent

[4] As initial value, one can consider the minimum possible reward, which is computed using the links' free flow travel times (as described in Section 2.1). From a practical perspective, such information could be easily obtained using any offline navigation device.

Table 1: Performance of our approach in different road networks. For each tested network, we show: the average travel time (which, ideally, should approximate the UE), the UE (as reported in the literature), the average estimated agent regret (Equation (7)), the average real agent regret (Equation (4)), and the relative difference between the estimated and real agent regrets.

network	*avg-tt*	UE	estimated regret	real regret	relative difference (%)
`Pigou`	1.000	1.000	0.0136	0.0135	4.11
`OW`	67.156	67.157	0.0031	0.0045	8.02
`1st Braess graph`	1.988	2.000	0.0245	0.0224	8.25
`2nd Braess graph`	2.832	3.000	0.0393	0.0221	41.66
`3rd Braess graph`	3.701	4.000	0.0882	0.0293	64.64

regret (using Equation (7)), both regarding the last episode. We tested different combinations for the Q-learning parameters. For each such combination, 30 repetitions were performed. The best[5] combination found was $\alpha = 0.5$, $\epsilon = 1.0$ and $\lambda = 0.99$. Moreover, in the case of the `OW` network, we also tested different values for k (the KSP algorithm is run once for each OD pair, in the beginning of the experiment). The best results were achieved for $k = 8$. The results of such configurations were selected for further analysis in the next subsection.

The main hypotheses of our work are that: (i) the results approach the user equilibrium (UE), and (ii) the regret estimations are reasonably precise. In the next subsection, we analyse how successful our approach performed regarding our initial hypotheses.

5.2 Results

The performance of our approach in all tested road networks is presented in Table 1. In the table, we show the two main performance metrics *avg-tt* and average estimated agent regret. Additionally, in order to analyse such results, we present the UE (as reported in the literature), the average real agent regret (using Equation (4)[6]), and the relative difference between the estimated and real regrets.

Our first hypothesis states that the *avg-tt* results are close to the UE. As shown in Table 1, such results are indeed close to the UE values reported in the literature. The results become slightly far from the UE for the Braess graphs, especially the larger ones ($p > 1$). This phenomenon can be explained by the nature of such graphs. Under UE, only the so-called type-P routes are used (see Appendix A for a detailed description). However, such routes have very similar costs. Consequently, it becomes harder for the agents to choose which route to take. The problem becomes even harder for larger Braess graphs because the number of type-P routes also increases with p. Furthermore, the Braess graphs were designed so that the UE values are the farthest possible from the System Optimum (SO). Nonetheless, observe that, for these particular graphs, our *avg-tt* results are closer to the SO than those of the UE. Therefore, such results evidence that our approach provides good approximations to the UE.

Regarding our second aforementioned hypothesis, its validation involves evaluating how precise the regret estimations are. To analyse such precision, we compare the real and estimated agent regrets by means of their relative difference. The lower such difference, the better. Of course, such difference cannot be computed by the agents, otherwise the regret estimation would not be necessary. As can be seen in Table 1, the estimated regrets are reasonably close to the real ones, especially for the networks `Pigou`, `OW` and `1st Braess graph`. For the larger Braess graphs, the results were somewhat worse. The point here, again, is that the Braess graphs are more challenging because they were designed to be among the hardest networks. As the agents have more difficulty to learn their best routes, the network takes longer to reach a more stable point. Consequently, the agents estimations need to be updated more frequently to account for the high variations in the routes costs. However, despite the difficulties, the estimations were reasonable. We highlight that such estimations could be greatly improved by adopting more sophisticated estimation methods (e.g., using a nonlinear regressor). Thus, the present results also evidence that our regret estimation mechanism reaches a reasonable level of precision.

Along these lines, we can conclude, at least experimentally, that the agents can, in fact, estimate their regret locally and use such information to learn their best routes. Obviously, these results are not definitive. As initially mentioned, this work represents a very first step towards a more formal investigation regarding formal guarantees for RL algorithms, which is our medium-term objective.

6 Concluding Remarks

In this paper, we addressed the route choice problem by minimising the drivers' regret. This problem concerns how rational drivers learn which route to take so as to minimise their expected travel costs. To this regard, we developed methods for learning agents to estimate their regret locally (i.e., based exclusively on their experience) and to learn using such estimations. Specifically, each agent keeps a history of experimented rewards, which is used to compute the so-called estimated action regret.

[5]The best value for α varied slightly from one network to another. However, such a value was reasonably close to 0.5 in all tested cases. Thus, for uniformity, we chose $\alpha = 0.5$ for all networks.

[6]In order to compute the real regret of Equation (4), we considered that the space of policies consists of a simple mapping from routes to deterministic policies. In fact, ignoring mixed policies over the set of available actions is a common practice in the literature [Banerjee and Peng, 2005; Zinkevich *et al.*, 2008].

Based on experiments, we have shown that our approach provides reasonably precise estimations of the agents' regret and that such estimations are useful in the learning process. We recall that this work represents an initial effort towards a more formal investigation of efficiency guarantees for RL algorithms, which is our medium-term objective.

For future work, we would like to investigate formally how precise the regret estimations might be. We expect that more sophisticated methods could be employed to estimate the agents' regret (e.g., using a nonlinear regressor). Moreover, we would like to study how much our approach benefits when the agents can communicate to improve their estimations. We also consider investigating the benefits of employing an online service for providing global information for the agents. Regarding the learning process, a promising direction would be adopting algorithms that learn mixed policies over the actions rather than only the best action. Furthermore, considering this work is a proof-of-concept, no comparison was made against other methods in the literature. Thereby, making such a comparison is an obviously important step to provide a more complete analysis of our approach. Last but not least, it would be interesting to explore how the learning process could be shaped towards globally efficient routes.

Acknowledgments

We are very grateful to the anonymous reviewers for their valuable comments. The authors are partially supported by CNPq and CAPES grants.

References

[Banerjee and Peng, 2005] Bikramjit Banerjee and Jing Peng. Efficient no-regret multiagent learning. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, pages 41–46. AAAI Press, 2005.

[Bazzan and Klügl, 2013] Ana L. C. Bazzan and Franziska Klügl. *Introduction to Intelligent Systems in Traffic and Transportation*, volume 7 of *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan and Claypool, 2013.

[Bowling and Veloso, 2002] Michael Bowling and Manuela Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250, 2002.

[Bowling and Zinkevich, 2012] Michael Bowling and Martin Zinkevich. On local regret. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pages 1631–1638, New York, USA, 2012. Omnipress.

[Braess, 1968] D. Braess. Über ein Paradoxon aus der Verkehrsplanung. *Unternehmensforschung*, 12:258, 1968.

[Buşoniu *et al.*, 2008] L. Buşoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(2):156–172, 2008.

[Hannan, 1957] James Hannan. Approximation to Bayes risk in repeated play. *Contributions to the Theory of Games*, 3:97–139, 1957.

[Koutsoupias and Papadimitriou, 1999] Elias Koutsoupias and Christos Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th annual conference on Theoretical aspects of computer science (STACS)*, pages 404–413, Berlin, Heidelberg, 1999. Springer-Verlag.

[Ortúzar and Willumsen, 2011] Juan de Dios Ortúzar and Luis G. Willumsen. *Modelling transport*. John Wiley & Sons, Chichester, UK, 4 edition, 2011.

[Pigou, 1920] A. Pigou. *The Economics of Welfare*. Palgrave Classics in Economics. Palgrave Macmillan, 1920.

[Powers and Shoham, 2005] Rob Powers and Yoav Shoham. New criteria and a new algorithm for learning in multi-agent systems. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1089–1096. MIT Press, 2005.

[Roughgarden, 2006] Tim Roughgarden. On the severity of Braess's paradox: Designing networks for selfish users is hard. *Journal of Computer and System Sciences*, 72(5):922–953, 2006.

[Shoham *et al.*, 2007] Yoav Shoham, Rob Powers, and Trond Grenager. If multi-agent learning is the answer, what is the question? *Artificial Intelligence*, 171(7):365–377, May 2007.

[Stone and Veloso, 2000] Peter Stone and Manuela Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383, July 2000.

[Sutton and Barto, 1998] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[Wardrop, 1952] John Glen Wardrop. Some theoretical aspects of road traffic research. In *Proceedings of the Institution of Civil Engineers*, volume 1, pages 325–362, 1952.

[Watkins and Dayan, 1992] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.

[Waugh *et al.*, 2015] Kevin Waugh, Dustin Morrill, J Andrew Bagnell, and Michael Bowling. Solving games with functional regret estimation. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2138–2144. AAAI Press, 2015.

[Yen, 1971] Jin Y. Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17(11):712–716, 1971.

[Zinkevich *et al.*, 2008] Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. In J C Platt, D Koller, Y Singer, and S T Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1729–1736. Curran Associates, Inc., 2008.

[Zinkevich, 2003] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *In Proceedings of the Twentieth International Conference on Machine Learning*, pages 928–936, Menlo Park, USA, 2003. AAAI Press.

A Expanding the Braess Graphs

The original Braess graph was designed to illustrate the counter-intuitive phenomenon that removing a link from a road network may improve its outcome [Braess, 1968]. This is the so-called Braess paradox. In this paper, we are not interested in the paradox itself. However, we employed the Braess graph for validating our approach given it poses some interesting challenges in the drivers' decision process (as seen in Section 5.2). Roughgarden devised a method for generating the p^{th} expansion of the original Braess graph [Roughgarden, 2006]. Nonetheless, the proposed modelling required the flow (i.e., the number of vehicles) to be normalised in the interval $[0, p]$, with p the order of the Braess graph. In order to overcome such limitation, we extend Roughgarden's modelling dropping such a requirement, thus delivering a more general model. Moreover, we provide updated theoretical results for the System Optimal (SO) and User Equilibrium (UE) solution concepts, as well as the results for the Braess paradox and the price of anarchy.

A.1 Graphs Generation Process

Consider the modelling introduced in Section 2.1. Let B^p be the p^{th} Braess graph, with B^1 being equivalent to the original Braess graph. The set of nodes can be described as $N^p = \{s, n_1, \ldots, n_p, o_1, \ldots, o_p, t\}$, with $|N^p| = 2p+2$ and $s \in N$ and $t \in N$ representing the source and target nodes, respectively, for all d drivers (i.e., all drivers share the same OD pair). Let (i, j) represent a link from $i \in N$ to $j \in N$. The set of links can be formulated as $L^p = \{(s, n_i), (n_i, o_i), (o_i, t) : 1 \leq i \leq p\} \cup \{(n_i, o_{i-1}) : 2 \leq i \leq p\} \cup \{(n_1, t), (s, o_p)\}$, with $|L^p| = 4p + 1$. The links are grouped into three distinct types, each with a corresponding cost function, as follows.

type-A : for all links on the form (n_i, o_i), we use $c_l^p(f_l) = 0$;

type-B : for all links on the form (n_i, o_{i-1}), (s, o_p), and (n_1, t), we use $c_l^p(f_l) = 1$;

type-C : for all links on the form (o_i, t) and (s, n_{p-i+1}), with $i \in \{1, \ldots, p\}$, we use a piecewise, continuous, non-decreasing function as in Equation (8). Using this function, the maximum possible cost of a type-C link (when $f_l = d$) is ip^2. The shape of the type-C cost function is illustrated in Figure 1.

$$c_l^p(f_l) = \begin{cases} 0 & \text{if } f_l \leq d/(p+1) \\ \frac{ip(pf_l + f_l - d)}{d} & \text{otherwise} \end{cases} \quad (8)$$

The routes are divided into two groups. Let P denote[7] the set of routes without any type-C link, i.e., $P = \{P_i = (s, n_i, o_i, t) \mid i \in [1, p]\}$, with $|P| = p$. All the other routes, with type-C links, are then represented by $Q = \{(s, n_1, t)\} \cup \{Q_i = (s, n_i, o_{i-1}, t) \mid i \in [2, p]\} \cup \{(s, o_p, t)\}$, with $|Q| = p + 1$. We will distinguish the routes from these two groups as type-P and type-Q routes.

[7] We slightly abuse notation here, using $(s, \ldots, t)$ to represent a sequence of connected links $\{(s, \cdot), \ldots, (\cdot, t)\}$ that form a route.

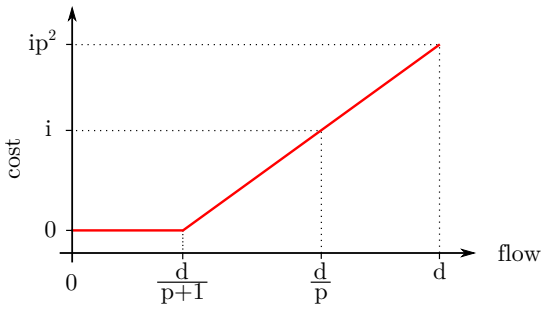
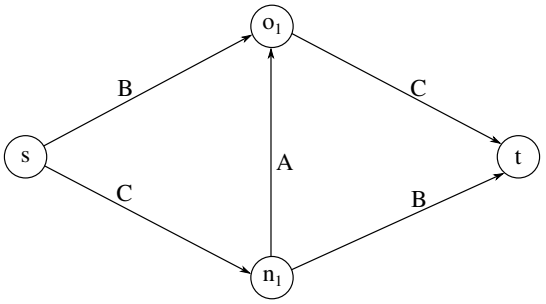
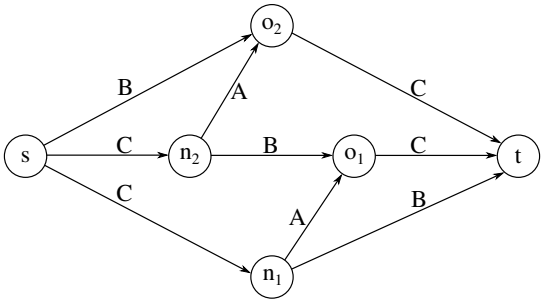


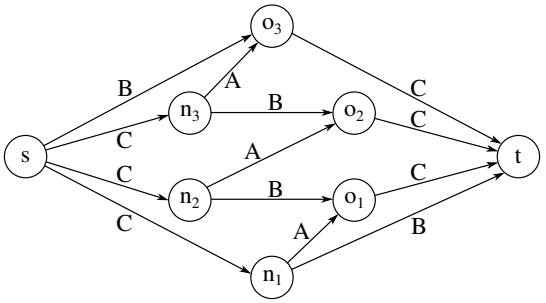
Figure 1: Shape of type-C cost functions.



(a) `1st Braess graph`



(b) `2nd Braess graph`



(c) `3rd Braess graph`

Figure 2: The first, second, and third Braess graphs. Links are labelled with their types.

A.2 Theoretical Results

Given the above formulation, we can define theoretical results for the System Optimum (SO) and the User Equilibrium (UE) as follows. The SO is achieved when the total flow d is equally divided among the $p + 1$ type-Q routes. In this case,

each such route receives a portion $d/(p+1)$ of the flow and the type-P routes are not used. Under such conditions, each route has a cost of 1 and the *avg-tt* is also 1.

The UE, on the other hand, is achieved when only the p type-P routes are used. In this case, each such route receives a flow of d/p, thus costing $p+1$. Under such circumstances, the *avg-tt* is also $p+1$. By comparing the SO and UE results, we can define the price of anarchy to be $p+1$ [Koutsoupias and Papadimitriou, 1999].

Observe that, in both solution concepts, the *avg-tt* and the route costs are always the same. This is due to the fact that, in both cases, all routes being used have precisely the same flow and cost, i.e., under SO all used routes have a flow of $d/(p+1)$ and cost 1 each, and under UE all used routes have a flow of d/p and cost $p+1$. Consequently, as all vehicles experiment the same cost, we have that the *avg-tt* and the route costs are always the same.

Regarding the Braess paradox, our modelling does not invalidate it. Observe that, whenever the type-A links (those in the form (n_i, o_i)) are removed, all type-P routes are also eliminated. Moreover, recall that, under UE, the cost of the flow is $p+1$. However, after the type-P routes are removed, the UE is achieved when the flow is equally divided among the type-Q routes, which is precisely the SO. Thus, removing the type-P routes leads to an improvement in the overall performance, meaning that the Braess paradox exists.