UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

GABRIEL DE OLIVEIRA RAMOS

Regret Minimisation and System-Efficiency in Route Choice

Thesis presented in partial fulfillment
of the requirements for the degree of
Doctor of Computer Science

Advisor: Prof. Dr. Ana Lúcia Cetertich Bazzan

Porto Alegre
March 2018

CIP — CATALOGING-IN-PUBLICATION

Ramos, Gabriel de Oliveira

Regret Minimisation and System-Efficiency in Route Choice / Gabriel de Oliveira Ramos. – Porto Alegre: PPGC da UFRGS, 2018.

148 f.: il.

Thesis (Ph.D.) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2018. Advisor: Ana Lúcia Cetertich Bazzan.

1. Multiagent reinforcement learning. 2. Route choice. 3. User equilibrium. 4. System optimal. 5. Regret minimisation. 6. Action regret. 7. Travel information. 8. Marginal-cost tolling. I. Bazzan, Ana Lúcia Cetertich. II. Título.

To my family.

“Don’t only practice your art, but force your way into its secrets, for it and knowledge can raise men to the divine.”

— Ludwig van Beethoven

ACKNOWLEDGEMENTS

This thesis is the result of years of intense study, persistence, poorly slept nights, and so on (if you are reading this, you know how long this list is). In spite of these challenges, I believe the result has paid off. Nonetheless, I could not take credit for this all alone. In fact, this work was only made possible thanks to several people who stood by my side and thus contributed in different ways for the development of my research.

First of all, I would like to express my deepest gratitude to my advisor, Prof. Ana Bazzan, for sharing her knowledge with me and for showing me the way of my research. Thank you for your patience, for the valuable advice after uncountable reviewing iterations, and for all growth opportunities offered throughout my journey. I also would like to express profound recognition to my co-advisor, Prof. Bruno Castro, for his critical (though friendly) feedback, and for the time spent on meetings and reviewing papers. Furthermore, thank you for putting such an effort on advising me even in the absence of official recognition. Thank you both for being such an example of academic excellence.

This work would not take place without the unconditional support of my family: Adilson, Sirlei, and Guilherme. I have no words to describe how blessed I am for their love, for the immeasurable patience and comprehension, for (emotionally and financially) supporting me, for looking after me, for comforting me, for understanding my absence, for trusting me, and for believing in me. You are an essential part of my life and I love you all. To my (now) wife Daiane, besides all above, I also thank her for giving me the honour to be part of her life, for accompanying me in my journey, and for postponing her dreams in favour of mine. I would be nothing without you all. Thank you!

I also thank my dearest friends from INF-UFRGS. Anderson, for his patience and for his inspiring advice. Daniel, for showing me that what comes next is even more exciting. Mariana, for her kind and wise advice. Renan, for the relaxing talks during the otherwise stressful daily trips. Ricardo, for the encouraging, insightful discussions on the hardest problems. I also thank Alejandro, Arthur, Andrew, Diego, Fernando, Jorge, Liza, Sérgio, Rafael, Thiago, and so many others that I cannot remember here (sorry!). Above all, for them and all others, thanks for being my friends and for helping me survive the intricate graduate life!

I could not forget to thank INF's professors, for being such an inexhaustible source of knowledge. Furthermore, I am indebted to all INF's staff: Alexsander, Cláudia, Eliane, Elisiane, Elizabeth, Flávia, Jorge, Leandro, Luis Otávio, Silvânia, and several others. My

way until here would be much harder without your dedication.

In the last year of my PhD, I was fortunate to spend a month at VUB AI-Lab with Prof. Ann Nowé. I thank her for receiving me in Brussels and for giving me the opportunity to further develop my research. I also thank the other members of her lab, especially Arno, Elias, Roxana, and Tim, for their kind support during my stay, for the lunches, and for the social activities. Thank you all!

My journey was way enjoyable due to supportive advice from other researchers. One such person is Prof. Juan Burguillo, with whom I have been collaborating since my masters. Thank you for being such a nice person and for incentivising me to keep on my way. I also thank Prof. Francesco Amigoni, for mentoring me during AAMAS 2017.

I also thank the examination committee of my thesis: Prof. Ann Nowé, Prof. Anna Reali, Prof. Felipe Meneguzzi, and Prof. Luciana Buriol. Thank you for the thorough review and for the helpful, constructive feedback.

Finally, this work would not be made possible without the financial support of several funding agencies. I thank the Brazilian agencies CNPq and CAPES for my PhD scholarship. I also thank ACM, EurAI, FAPERGS, IFAAMAS, SIGAI, SIGEVO, and PROPESQ for supporting my travels to conferences and summer schools, which contributed towards my formation.

ABSTRACT

Multiagent reinforcement learning (MARL) is a challenging task, where self-interested agents concurrently learn a policy that maximise their utilities. Learning here is difficult because agents must adapt to each other, which makes their objective a moving target. As a side effect, no convergence guarantees exist for the general MARL setting. This thesis exploits a particular MARL problem, namely *route choice* (where selfish drivers aim at choosing routes that minimise their travel costs), to deliver convergence guarantees. We are particularly interested in guaranteeing convergence to two fundamental solution concepts: the user equilibrium (UE, when no agent benefits from unilaterally changing its route) and the system optimum (SO, when average travel time is minimum).

The main goal of this thesis is to show that, in the context of route choice, MARL can be guaranteed to converge to the UE as well as to the SO upon certain conditions. Firstly, we introduce a regret-minimising Q-learning algorithm, which we prove that *converges to the UE*. Our algorithm works by estimating the regret associated with agents' actions and using such information as reinforcement signal for updating the corresponding Q-values. We also establish a bound on the agents' regret. We then extend this algorithm to deal with non-local information provided by a navigation service. Using such information, agents can improve their regrets estimates, thus performing empirically better. Finally, in order to mitigate the effects of selfishness, we also present a generalised marginal-cost tolling scheme in which drivers are charged proportional to the cost imposed on others. We then devise a toll-based Q-learning algorithm, which we prove that *converges to the SO* and that is fairer than existing tolling schemes.

Keywords: Multiagent reinforcement learning. Route choice. User equilibrium. System optimal. Regret minimisation. Action regret. Travel information. Marginal-cost tolling.

Minimização de Regret e Eficiência do Sistema em Escolha de Rotas

RESUMO

Aprendizagem por reforço multiagente (do inglês, MARL) é uma tarefa desafiadora em que agentes buscam, concorrentemente, uma política capaz de maximizar sua utilidade. Aprender neste tipo de cenário é difícil porque os agentes devem se adaptar uns aos outros, tornando o objetivo um alvo em movimento. Consequentemente, não existem garantias de convergência para problemas de MARL em geral. Esta tese explora um problema em particular, denominado escolha de rotas (onde motoristas egoístas deve escolher rotas que minimizem seus custos de viagem), em busca de garantias de convergência. Em particular, esta tese busca garantir a convergência de algoritmos de MARL para o equilíbrio dos usuários (onde nenhum motorista consegue melhorar seu desempenho mudando de rota) e para o ótimo do sistema (onde o tempo médio de viagem é mínimo).

O principal objetivo desta tese é mostrar que, no contexto de escolha de rotas, é possível garantir a convergência de algoritmos de MARL sob certas condições. Primeiramente, introduzimos uma algoritmo de aprendizagem por reforço baseado em minimização de arrependimento, o qual provamos ser capaz de convergir para o equilíbrio dos usuários. Nosso algoritmo estima o arrependimento associado com as ações dos agentes e usa tal informação como sinal de reforço dos agentes. Além do mais, estabelecemos um limite superior no arrependimento dos agentes. Em seguida, estendemos o referido algoritmo para lidar com informações não-locais, fornecidas por um serviço de navegação. Ao usar tais informações, os agentes são capazes de estimar melhor o arrependimento de suas ações, o que melhora seu desempenho. Finalmente, de modo a mitigar os efeitos do egoísmo dos agentes, propomos ainda um método genérico de pedágios baseados em custos marginais, onde os agentes são cobrados proporcionalmente ao custo imposto por eles aos demais. Neste sentido, apresentamos ainda um algoritmo de aprendizagem por reforço baseado em pedágios que, provamos, converge para o ótimo do sistema e é mais justo que outros existentes na literatura.

Palavras-chave: Aprendizagem por reforço multiagente, Escolha de rotas, Equilíbrio dos usuários, Ótimo do sistema, Minimização de regret, Regret da ação, Informação de viagem, Pedágio de custo marginal.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS AND ACRONYMS

avg-tt	average travel time
BPR	Bureau of Public Roads
FFQ	Friend-or-Foe-Q (algorithm)
GIGA	Generalized Infinitesimal Gradient Ascent (algorithm)
IGA	Infinitesimal Gradient Ascent (algorithm)
ITS	Intelligent Transportation Systems
KSP	K Shortest Loopless Paths (algorithm)
MARL	Multiagent Reinforcement Learning
MCT	Marginal-Cost Tolling
MDP	Markov Decision Process
NE	Nash Equilibrium
OD	Origin-Destination (pair)
OW	Ortúzar and Willumsen (an instance of the route choice problem)
PoA	Price of Anarchy
RL	Reinforcement Learning
RRM	Random Regret Minimisation
RT	Regret Theory
SF	Sioux Falls (an instance of the route choice problem)
SO	System Optimum
stdQL	standard Q-learning
TD	Temporal Difference
UE	User Equilibrium
VDF	Volume-Delay Function
WoLF	Win or Learn Fast (algorithm)
WPL	Weighted Policy Learner (algorithm)

LIST OF SYMBOLS

α	learning rate (also a constant of the BPR function)
β	constant of the BPR function
ϵ	exploration rate
λ	decay rate of α
μ	decay rate of ϵ
μ^t	value of μ at time t
ϕ	approximation rate of the UE (also regret bound)
π	reinforcement learning policy
$\rho(\overset{+}{a}{}_i^t)$	(also $\overset{+}{\rho}{}_i^t$, $\overset{+}{\rho}$) probability that agent i selects its best action at time t
$\rho(\bar{a}_i^t)$	(also $\bar{\rho}_i^t$, $\bar{\rho}$) probability that agent i selects a non-best action at time t
τ_l	toll on link l
$\mathcal{A}$	set of actions (also a joint action space)
A_i	set of actions of agent i
a_i^t	*a* route of agent i at time t
$\dot{a}_i^t$	route actually taken by agent i at time t
$\overset{*}{a}{}_i^t$	true best action (that with true highest reward) of agent i at time t
$\overset{+}{a}{}_i^t$	best action (that with highest Q-value) of agent i at time t
$\bar{a}_i^t$	any non-best action (one that does not have the highest Q-value) of agent i at time t
B^p	p^{th} Braess graph
BB^p	p^{th} Bi-commodity Braess graph
C_R	cost of route R
C_l	capacity of link l (only used in the BPR function)
c_l	cost on link l
D	set of driver agents

d	number of drivers, i.e., $d = \|D\|$
F_l	free flow travel time on link l
f_l	travel time on link l (VDF function)
f'_l	derivative of the VDF function for link l
G	graph representation of a road network
H_i	history of reward estimates of agent i
P	instance of the route choice problem
K	number of available routes
L	set of links in the graph
l	number of links
m	number of OD pairs
N	set of nodes in the graph
n_u	node u
$\mathcal{P}$	set of players (in the context of Markov games)
p'_1	constant of the VDF
Q	Q-value
$Q(a)$	Q-value of action a
R	route
$\mathcal{R}_i^T$	external regret of agent i up to time T
$\tilde{\mathcal{R}}_i^T$	estimated external regret of agent i up to time T
$\tilde{\mathcal{R}}_{i,a}^T$	estimated action regret of agent's i action a up to time T
$r(s, a)$	reward received after taking action a in state s
$r(\dot{a}_i^t)$	reward received by agent i at time t for taking action $r(\dot{a}_i^t)$
$\tilde{r}(a_i^t)$	most recent reward *estimate* of agent i for taking action a on time t
$\hat{r}(a^t)$	average reward of action a up to time t (as compute by the app)
$\mathcal{S}$	set of environment states

$\mathcal{T}$	transition function
T	time (episode) in the limit
t	time (episode)
x_l	flow on link l

CONTENTS

1 INTRODUCTION

Efficient urban mobility plays a major role in modern societies. Notwithstanding, the fast-growing demand for mobility associated with the lack of appropriate investments has compromised the efficiency of traffic systems, as evidenced by the increasing number (and intensity) of traffic congestions. In fact, according to the Centre for Economics and Business Research (2014), the cost imposed by traffic congestions on the economy of the USA was around US$ 120 billion in 2013. Furthermore, as suggested by the same report, such costs will increase by 50% until 2030.

Traditional approaches for dealing with arising traffic congestions include increasing the physical capacity of existing traffic infrastructure. Such approaches, nonetheless, have proven unsustainable from many perspectives (e.g., economic, environmental). Furthermore, as stated by the Braess (1968)'s paradox, expanding the infrastructure's capacity may even deteriorate the traffic performance.

Against this background, ways of making a more efficient use of the existing infrastructure have been increasingly studied. In fact, the increasing cooperation of the traffic engineering and computer science fields has brought insightful results. In particular, the use of artificial intelligence has leveraged the development of the so-called intelligent transportation systems (ITS), which aim at promoting the use of technology to gather and integrate information in order to improve the efficiency of the transportation system (BAZZAN; KLÜGL, 2013). In the literature, several works put an effort on modelling traffic as an optimisation problem and indeed succeeded in mitigating congestion effects. However, recent advances in information and telecommunication paved the way for autonomous, distributed solutions for dealing with traffic issues.

In this thesis, we approach traffic from the drivers' viewpoint and investigate how they decide on which route to take everyday. Observe that drivers' decisions are intrinsically self-interested (i.e., regarding their own benefit) and affect the way other drivers perceive traffic. In this context, explicitly saying which route a driver should choose in what situation becomes pointless. Facing such challenges, we are interested in investigating how drivers can effectively *learn* to make their decisions based on their previous experiences. We can then approach the problem from the reinforcement learning perspective. By proceeding this way, we look forward to delivering simple traffic solutions that, in a near future, could be easily deployed to enhance traffic as perceived by drivers.

1.1 Motivation

Reinforcement learning (RL) in multiagent domains is a challenging task. An RL agent must learn by trial-and-error how to behave within its environment in order to maximise its utility. Precisely, the agent aims at learning an optimal behaviour. In the basic, single-agent RL setting, several algorithms are guaranteed to converge to such an optimal behaviour (KAELBLING; LITTMAN; MOORE, 1996). However, when multiple, self-interested agents share a common environment, their utilities may be affected by each others' decisions (LITTMAN, 1994; CLAUS; BOUTILIER, 1998; BUŞONIU; BABUSKA; SCHUTTER, 2008). In this regard, agents must adapt their behaviour to each other, which makes their objective a moving target. Due to such dynamics, no convergence guarantees exist for the general multiagent RL (MARL) setting, i.e., for an arbitrary number of players and actions. In order to overcome such limitations, literature on MARL has been focused on exploiting the structure of specific problems to investigate convergence guarantees. In this thesis, we follow such a direction by considering a particular MARL problem, namely the *route choice problem*, and exploit its structure to deliver convergence guarantees.

The route choice problem concerns how self-interested drivers (agents[1]) behave when choosing routes (actions) between their origins and destinations in order to minimise their travel costs (e.g., time, money). Whenever a driver takes a route, it affects the traffic conditions as perceived by other drivers. Consequently, learning plays a role in such situation, since the agents must adapt their choices to account for the changing traffic conditions. Thus, the route choice problem presents a challenging scenario for MARL.

In general terms, the performance of route choice can be described considering both individual and global aspects. In this sense, the most typically studied solution concepts are the user equilibrium (UE) and the system optimum (SO) (WARDROP, 1952). The UE is achieved when no driver benefits from unilaterally changing its route. As such, the UE can be seen as a consequence of the agents' self-interested behaviour. The SO, on the other hand, represents the system at its best operation (i.e., when the average travel cost is minimum), and is only attainable if some agents take sub-optimal routes in favour of the system's performance. We emphasise, nonetheless, that considering that agents act rationally to minimise their own costs, it is not realistic to assume that they will take routes that would lead to the SO: whenever a better route is available, the agents shall pre-

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

fer it. Such a deterioration in the system's performance due to drivers' selfish behaviour is known as the Price of Anarchy (PoA) (PAPADIMITRIOU; TSITSIKLIS, 1987).

1.2 Research question and challenges

Motivated by the above discussion, this thesis is driven by the following question: in the context of the route choice problem, is it possible to design a reinforcement learning algorithm that is guaranteed to converge to the user equilibrium or to the system optimum? In the MARL literature, some works have partially answered the this question. However, existing guarantees are limited to specific cases and do not apply to the route choice problem (a discussion on this topic is presented in Chapter 2).

In this thesis, we are interested in analysing how RL agents can learn on their own with performance guarantees, and dropping usual assumptions made in the literature (such as that agents have full knowledge about the reward functions, which we discuss next). In this regard, we identify three challenges *in the context of route choice* that need to be addressed to achieve the desired convergence guarantees. In order to enhance presentation, we first formulate each challenge in general terms (as a question) and then we delve into its details (presenting the possible means to handle it).

- *Under what conditions can we guarantee that RL agents will converge to the UE?* As discussed above, when several agents compete for a common resource, the learning objective becomes a moving target. This is the case of route choice, where learning means finding the best route to take, and agents' decisions affect the reward received by others. In this regard, establishing a bound on the agents' performance becomes challenging. The use of regret-minimising algorithms has shown promising results (CESA-BIANCHI; LUGOSI, 2006). Roughly, regret measures how much worse an agent performs on average in comparison to the best fixed action in hindsight. Some progress has been made by employing regret in the context of RL (BOWLING, 2005; ZINKEVICH et al., 2008; WAUGH et al., 2015), congestion games (BLUM; EVEN-DAR; LIGETT, 2010), multi-armed bandits (AUER et al., 2002; AWERBUCH; KLEINBERG, 2004). Nonetheless, most works assume that the agents (or a central authority) have full knowledge (about the cost functions) and can compute their regret. Hence, the challenge here is twofold: providing means for the agents to estimate their regret locally

(i.e., based exclusively on their experience), and ensuring that such regret-based MARL approach converges to the UE.

- *Under what conditions can we enhance the agents' learning process using non-local information?* Building upon the previous question, a natural extension regards investigating how the provision of travel information affects the agents' learning process. Considering the increasing adoption of mobile navigation devices, the impact of such devices on the system's performance should not be neglected (MITCHELL; BORRONI-BIRD; BURNS, 2010; NEW CITIES FOUNDATION, 2012). When such devices are available, the provided information may be incorporated into the agents' regret (e.g., agents may regret for taking—or not—a suggested route). In this sense, combining an agent's experience (local information) with information provided by a navigation device (non-local information) represents a promising direction. In general, however, existing works assume either that agents have full knowledge about the cost functions (BEN-ELIA; ISHAQ; SHIFTAN, 2013) or that a central authority (responsible for providing non-local information) can observe agents' actions a priori (KLÜGL; BAZZAN, 2004; VASSERMAN; FELDMAN; HASSIDIM, 2015). Therefore, the main challenges here refer to defining the nature of the non-local information (i.e., so that full knowledge is not required), and to effectively combining the received information with the agents' perceptions.
- *Under what conditions can we guarantee that RL agents will converge to the SO?* When agents seek to minimise their travel costs, the system converges to the UE. However, recall that the UE is inefficient from the system's perspective. Whereas agents cannot be enforced to behave altruistically (FEHR; FISCHBACHER, 2003), the use of tolls can achieve equivalent results (BECKMANN; MCGUIRE; WINSTEN, 1956). Building upon previous points, regret *could* be reformulated to account for the impact an agent causes on others. This is similar to tolling an agent proportionally to such impact, which is called marginal-cost tolling (MCT), as defined by Pigou (1920). We then concentrate on the case of tolls. It should be noted, nonetheless, that the existing literature on tolls usually assume that a full-knowledged central authority computes and charges such tolls (COLE; DODIS; ROUGHGARDEN, 2003; CHEN; KEMPE, 2008; SHARON et al., 2017), which relies on additional infrastructure. Thus, the challenge here concerns how to employ MCT in a decentralised way, without relying on full-knowledge assumptions.

1.3 Proposal and contributions

In this thesis, we investigate the convergence properties of MARL (regarding the UE and the SO) in the context of route choice. In this regard, we formulate the following hypotheses to answer our research question: (i) the use of regret as reinforcement signal leads reinforcement learning agents to converge to the user equilibrium, (ii) the provision of non-local information to the agents improves their learning performance, and (iii) the use of marginal-cost tolling leads reinforcement learning agents to converge to a system-efficient equilibrium[2] (i.e., the system optimum).

The main contributions of this thesis can be described as follows.

Learning from regret. We introduce a method through which agents can learn using their regret. Specifically, we show how agents can estimate their regret locally (i.e., based exclusively on their experience) and how such estimates can be employed to guide the RL process. In this regard, we introduce the notion of *action regret*, which measures the regret associated with every *single action*. Agents can *estimate* such action regret by keeping an internal history of observed rewards, thus eliminating any assumption of full knowledge. The action regret can then be used as reinforcement signal to update the agents' policies. We provide a theoretical analysis on the system's convergence, showing that our approach minimises the agents' regret and reaches an approximate UE. Moreover, we validate our theoretical analysis by means of experimental evaluation. These results also appear in Ramos, Silva and Bazzan (2017).

Using non-local information. We extend the above topic to deal with non-local information. Precisely, we present a method for the agents to estimate their regret using both local information (an internal history of observed rewards) and non-local information (provided by means of a mobile navigation entity, henceforth referred to as the *app*). The non-local information provided by the app is simply the average travel times of the routes used by the agents. In this sense, we reformulate an agent's action regret as a linear combination of its experience (rewards received in previous episodes) and information provided by the app. We perform

[2]Hereinafter, we employ the term *system-efficient equilibrium* to refer to an UE aligned to the SO, i.e., an equilibrium point that is no longer inefficient from the system's perspective. In such way, a system-efficient equilibrium is clearly equivalent to the SO (i.e., the average travel times in both cases is minimum). The distinction is necessary, however, to emphasise that in the former agents do not have any incentive to deviate, whereas in the latter they do have such an incentive.

an experimental evaluation, showing that the use of app-based information improves the agents' performance. These results are also reported in Ramos, Bazzan and Silva (2018).

Finding system-efficient equilibria. We present a toll-based mechanism through which drivers converge to a system-efficient equilibrium. We design tolls using the marginal-cost tolling (MCT) scheme, where the cost of a link comprises two terms: the travel time and the toll charged on it. We generalise the toll values formulation for univariate, homogeneous polynomial cost functions, which comprises the most commonly-used cost functions in the literature. In contrast to other methods in the literature, we assume that tolls are charged a posteriori (i.e., at the end of each trip). Furthermore, our toll formulation allows each agent to compute the toll value it has to pay. In this sense, we can eliminate unnecessary information (i.e., agents only need to know their travel times and their routes' free flow travel time) and unrealistic assumptions (i.e., no additional infrastructure is required, since tolls can be computed and charged by mobile navigation devices). We provide theoretical results showing that, in the limit, our method converges to the UE and that, by using MCT, the UE corresponds to the SO. Thus, in the limit, the PoA achieves its best ratio. Furthermore, we show that our mechanism is fairer than a priori toll schemes.

The aforementioned contributions provide an answer to our initial question. In particular, by employing the proposed methods, it is possible to formally guarantee that RL agents will converge to the UE and to the SO. As a result, at least in the context of route choice, MARL can achieve convergence guarantees. It should be noted, however, that although this thesis focuses on a particular MARL problem, our approach is not necessarily limited in that regard. Our analyses may apply to other MARL problems as well. This topic is briefly discussed in the conclusions.

1.4 Publications

The research described in this thesis also appeared in a number papers.

- **Gabriel de O. Ramos**, Ana L. C. Bazzan, Bruno C. da Silva. Analysing the impact of travel information for minimising the regret of route choice. *Transportation Research Part C: Emerging Technologies*, v. 88, p. 257–271, Mar 2018.

- **Gabriel de O. Ramos**, Bruno C. da Silva, Ana L. C. Bazzan. Learning to minimise regret in route choice. In: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017)*. São Paulo: IFAAMAS, 2017. p. 846–855.
- **Gabriel de O. Ramos**. Minimising regret in route choice (doctoral consortium). In: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017)*. São Paulo: IFAAMAS, 2017. p. 1855–1856.
- **Gabriel de O. Ramos**, Ana L. C. Bazzan. On estimating action regret and learning from it in route choice. In: *Proc. of the Ninth Workshop on Agents in Traffic and Transportation (ATT-2016)*. New York: CEUR-WS.org, 2016. p. 1–8.

Some results of this thesis are also reported in papers that are still in preparation:

- **Gabriel de O. Ramos**, Bruno C. da Silva, Ana L. C. Bazzan. A regret-minimising approach for learning (system-efficient) equilibria in route choice. *Paper in preparation, title might change*.
- **Gabriel de O. Ramos**, Bruno C. da Silva, Ana L. C. Bazzan. Learning system-efficient equilibria in route choice using tolls. *Paper in preparation, title might change*.

Finally, previous works related to this thesis also include:

- **Gabriel de O. Ramos**, Ana L. C. Bazzan. Efficient local search in traffic assignment. In: *2016 IEEE Congress on Evolutionary Computation (CEC)*. Vancouver: IEEE, 2016. p. 1493–1500.
- **Gabriel de O. Ramos**, Ana L. C. Bazzan. Towards the user equilibrium in traffic assignment using GRASP with path relinking. In: *Proc. of the 2015 Conference on Genetic and Evolutionary Computation (GECCO)*. New York: ACM, 2015. p. 473–480.
- **Gabriel de O. Ramos**, Ricardo Grunitzki. An improved learning automata approach for the route choice problem. In: *Agent Technology for Intelligent Mobile Services and Smart Societies*. Springer Berlin Heidelberg, 2015, (CCIS, v. 498). p. 56–67.

1.5 Thesis outline

Besides the present introduction (Chapter 1), this thesis is organised into five chapters, as follows.

- Chapter 2 reviews the background on the topics relevant to this thesis.
- Chapter 3 introduces our investigations towards learning from regret.
- Chapter 4 extends the results of Chapter 3 to the case where non-local information is available to the agents.
- Chapter 5 focuses on our MCT scheme to bias agents' decisions towards a system-efficient equilibrium.
- Chapter 6 concludes with the final remarks and discussions.

2 BACKGROUND AND LITERATURE REVIEW

In this chapter, we present a brief overview on the topics relevant to the present thesis. These topics are divided into five groups: route choice (Section 2.1), reinforcement learning (Section 2.2), regret minimisation (Section 2.3), use of non-local information (Section 2.1.3), and system-efficient equilibria (Section 2.1.4).

2.1 Route choice problem

The route choice problem concerns how drivers behave when choosing routes between their origins and destinations (OD pair, henceforth). In this section, we introduce the basic concepts related to route choice. For a more comprehensive overview, the interested reader is referred to Bazzan and Klügl (2013) (for an agent-centred perspective) and Ortúzar and Willumsen (2011) (for a traffic engineering perspective).

2.1.1 Problem modelling

An instance of the route choice problem can be defined as a tuple $P = (G, D, f)$. Let $G = (N, L)$ be a directed graph representing a road network, where the set of nodes N represents the intersections and the set of links L represents the roads between intersections. An example graph is illustrated in Figure 2.1, with four nodes and five links. Each driver $i \in D$ (with $|D| = d$) has an OD pair, which corresponds to its origin and destination nodes. A trip is made by means of a route[1]

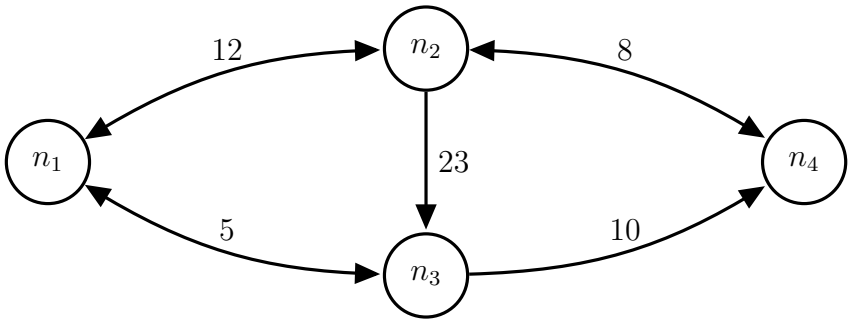
$$R = \left\{ \{n_u, n_v\} \in L \mid \forall p \in [0, |R| - 1], n_v^p = n_u^{p+1} \right\},$$

which is a sequence of links connecting an origin to a destination. For instance, in the example of Figure 2.1, the OD pair (n_4, n_3) has two possible routes (ignoring cycles), namely, $\{\{n_4, n_2\}, \{n_2, n_3\}\}$ and $\{\{n_4, n_2\}, \{n_2, n_1\}, \{n_1, n_3\}\}$. The demand for trips generates a flow of vehicles on the links, where x_l is the flow on link l (i.e., the *number of vehicles* using it). Each link $l \in L$ has a cost[2] $c_l : x_l \to \mathbb{R}^+$ associated with crossing it,

[1] We abuse notation here and use n_u^p (n_v^p) to denote the start (end) node of the p^{th} link of route R.

[2] In order to enhance presentation, we hereafter omit x_l from the definition of cost and travel time on link l, thus writing simply c_l and f_l rather than $c_l(x_l)$ and $f_l(x_l)$, respectively.

Figure 2.1: Graph representation of an example road network. The graph contains four nodes, representing intersections, and five links, representing roads. Links' labels represent the free flow travel time on the links. One-way roads are represented by unidirectional links and two-way roads by bidirectional links.



which is typically modelled as the travel time $f_l : x_l \to \mathbb{R}^+$ on it, i.e.,

$$c_l(x_l) = f_l(x_l) \tag{2.1}$$

The cost of a route R is then denoted by

$$C_R = \sum_{l \in R} c_l \tag{2.2}$$

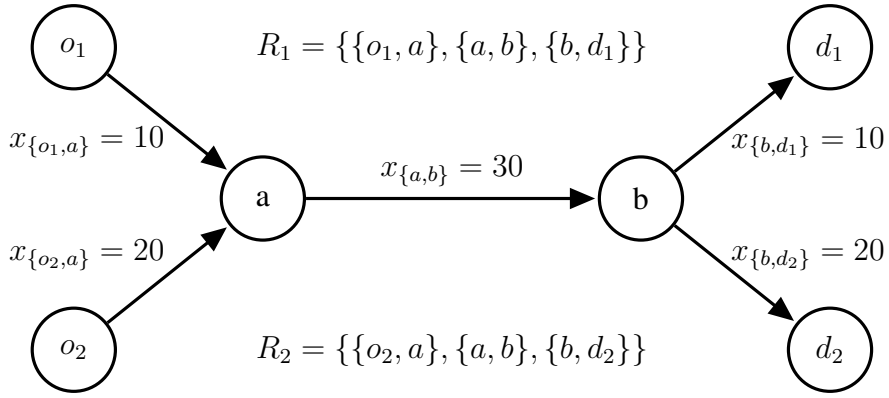
Travel times are typically abstracted by means of volume-delay functions (VDFs), which map the flow of vehicles on a link into the travel time on it (a.k.a. delay, usually measured in minutes). In order to correctly formulate a VDF, one needs to take several characteristics of the problem instance into account, which can be an arduous task. Fortunately, different general VDF formulations are available in the literature. In traffic engineering, the most used VDF was proposed by the American Bureau of Public Roads (1964) and is thus known simply as BPR. The BPR function is formulated as

$$f_l(x_l) = F_l \left(1 + \alpha \frac{{x_l}^{\beta}}{C_l}\right), \tag{2.3}$$

where F_l denotes the free flow travel time on link l (i.e., the minimum travel time, when the link is empty), C_l is the capacity[3] of link l, and α and β are constants (the same for all links) of the road network instance. Except for the flow (x_l), the other elements of the

[3]In the mathematical formulation of the route choice problem, the flow of vehicles on a link is not constrained to its capacity. Hence, VDFs are typically modelled as exponential functions to properly penalise overloaded links. We refer the reader to Ortúzar and Willumsen (2011) for a more detailed discussion on VDFs. A comparative analysis of the so-called macroscopic (i.e., our abstract mathematical formulation) and microscopic (i.e., a more detailed formulation, where even vehicles' speed, length, and position are considered in the simulation) models is presented in (BAZZAN; KLÜGL, 2013).

Figure 2.2: Example road network with two overlapping routes. In the example, the flow on link $\{a, b\}$ is obtained by summing up the flow on routes R_1 and R_2.



VDF are part of the road network instance and do not change along time. This includes F_l and C_l, which are fixed but each link has its own. A simpler, illustrative VDF is

$$f_l(x_l) = F_l + 0.02 \times x_l, \tag{2.4}$$

which increases the travel time on link l by 0.02 for each vehicle/hour of flow. We call this the OW function, due to Ortúzar and Willumsen (2011). It should be noted, however, that such a linear growth is not realistic: in general, traffic performance deteriorates faster at higher congestion rates. The BPR function, in contrast, defines an exponential growth for the delay, meaning that travel time increases faster as the flow of vehicles approaches the link's capacity.

At this point, considering that we name the problem as *route* choice, the eagle-eyed reader may ask why not simplifying the modelling by ignoring the links and considering just the routes themselves. In fact, this would be possible in simple scenarios, with disjoint routes. In general, however, routes tend to share at least some, but frequently *many*, links (e.g., arterial roads). Consequently, the flow of vehicles in a link is determined not only by a group of drivers using a single route that includes such link, but by groups of drivers using various routes that include such link at some point in their trips. This is illustrated in Figure 2.2, where the flow of vehicles on link $\{a, b\}$ is determined by the flow of vehicles using route R_1 and R_2. Therefore, links need to be considered in particular when modelling the route choice problem.

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 (i.e., the set of drivers

and their objectives do not change). Here, the time of the day is irrelevant and days are independent from each other. Each driver $i \in D$ can then be modelled as a reinforcement learning agent, which repeatedly deals with the problem of choosing the route (action) that takes the least time to its destination. In this case, the reward $r : R \rightarrow \mathbb{R}^+$ received by driver i after taking route R is inversely associated with the cost of such route, i.e.,

$$r(R) = -C_R. \tag{2.5}$$

We highlight that routes' costs may change from one day to another as a consequence of the drivers' adaptation to traffic conditions. This justifies the importance of learning within this scenario. More details are presented in Section 2.2.

The solution to the route choice problem is intuitively 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) and is equivalent to the Nash equilibrium (NASH, 1950). The UE is a consequence of the usual game-theoretic assumption that agents are rational and act selfishly. This way of behaving renders traffic as an intrinsically competitive environment. Observe, however, that the UE is inefficient from the system's perspective. In fact, the ideal outcome from the global perspective corresponds to the minimum average travel time, which is referred to as the system optimum (SO). The level of inefficiency due to selfish behaviour was formally defined by Papadimitriou and Tsitsiklis (1987) and ever since known as the Price of Anarchy (PoA). Roughly, the PoA is obtained by dividing the average travel time under the worst-case UE by that under SO. The minimum value for this ratio is 1, which is only possible if the UE is completely aligned to the SO.

Considering the inefficiency of the UE, there has been an increasing interest on minimising the PoA by biasing the UE towards the SO. Several alternatives have been proposed here (a more detailed discussion is presented in Section 2.1.4). In general, however, most approaches are equivalent to charging tolls on the links. In this regard, in this thesis we will refer to such variants as the *toll-based route choice problem*. An instance of the toll-based route choice problem is defined as $P = (G, D, f, \tau)$. The difference to the original route choice problem is that the cost[4] associated with crossing

[4]The usual formulation of the toll-based route choice problem defines the travel time and toll value as having the same weight. However, different weights could be easily defined by applying simple algebraic manipulations on the toll value itself.

link $l \in L$ is now given by

$$c_l(x_l) = f_l(x_l) + \tau_l(x_l), \tag{2.6}$$

where $f_l : x_l \to \mathbb{R}^+$ represents its travel time and $\tau_l : x_l \to \mathbb{R}^+$ denotes the *toll*[5] charged for using it.

Toll values can be defined according to different objectives (e.g., maximising revenue, minimising link usage). In this thesis, we are interested in biasing the UE towards the SO. According to Pigou (1920), this can be achieved by means of marginal cost tolling (MCT). Under MCT, each agent is charged proportionally to the cost it imposes on others. Specifically, the marginal cost toll on link l is the product of its flow (i.e., the number of vehicles on it) and the derivative of its VDF function (PIGOU, 1920), that is,

$$\tau_l = x_l \cdot (f_l(x_l))'.$$

As shown later, in Theorem 5.1, MCT aligns the UE to the SO. It should be noted, on the other hand, that charging tolls arbitrarily (e.g., charging a constant price on selected links) does not necessarily lead to the SO (BECKMANN; MCGUIRE; WINSTEN, 1956).

In this thesis, we concentrate our initial efforts towards finding the UE. In this regard, Chapters 3 and 4 address the basic route choice problem. Later on, Chapter 5 focuses on the convergence to a system-efficient equilibrium, in which case we use the toll-based variant of the problem.

2.1.2 Related problems

In this section, we present some problems that are analogous (or similar) to the route choice problems and comment on their differences. We highlight that this is a non-exhaustive list, limited to the most representative problems (at least for our purpose).

In the transportation literature, route choice is approached from different perspectives. *Discrete choice models* try to accurately approximate the behaviour of human travellers. An interesting overview of these methods is presented by McFadden (2001). *Assignment methods* are centralised mechanisms employed to find an allocation of vehicles into routes that satisfies a given solution concept, such as the UE. Examples of such mechanisms include Bar-Gera (2010) and Ramos and Bazzan (2015, 2016). We refer the reader to Sheffi (1984) and Ortúzar and Willumsen (2011) for a more thorough overview of these

[5] We hereafter write $\tau_l(x_l)$ simply as τ_l to enhance presentation.

and other perspectives. In general, however, these lines of research focus on facilitating the work of *traffic managers* in analysing different traffic patterns, policies, etc. On the other hand, in this thesis we consider the *drivers*' viewpoint. In particular, we investigate how self-interested driver-agents learn (with very limited knowledge) and adapt (considering that each agent's decisions affect other concurrently-learning agents) when trying to maximise their rewards. We also remark that our approach is completely distributed. For these reasons, our research is fundamentally different from that on discrete choice and assignment both in terms of the assumptions of which agents are the focus of the optimisation process (drivers or traffic managers), and in terms of the process by which the optimisation process occurs (centralised or distributed). The interested reader is referred to Bazzan and Klügl (2013) for a more detailed overview of traffic systems from the agents' perspective.

Congestion games (ROSENTHAL, 1973) represent another common way of approaching route choice. In fact, congestion games represent a generalisation of the route choice problem. In congestion games, the players' strategies consist of a multiset of resources (e.g., resources are links and strategies are routes), and the utility of a strategy depends only on the number of players using its resources (e.g., the congestion level on each of its links). We consider the specific case of (selfish) routing games (ROUGHGARDEN, 2005), in which strategies cannot be multisets of resources (after all, routes with cycles[6] are not reasonable from the route choice perspective). Routing games can be modelled as *atomic*, i.e., each player represents a commodity and controls its whole traffic, or *non-atomic*, i.e., each player controls a negligible, infinitesimally small amount of traffic (ROUGHGARDEN, 2007). The non-atomic model can be solved in polynomial time (BECKMANN; MCGUIRE; WINSTEN, 1956; FABRIKANT; PAPADIMITRIOU; TALWAR, 2004), whilst such theoretical results are weaker in the case of the atomic model (ROUGHGARDEN, 2005). The route choice problem can then be modelled as a non-atomic (selfish) routing game. We observe that, however, in our settings we have a finite set of players rather than an infinite set of infinitesimally small players (i.e., the flow is represented as real values). Furthermore, our primary focus is on how agents interact with each other and how their decisions affect each others' perceptions.

The multi-armed bandit problem (ROBBINS, 1952) can also be used to model route choice. In this problem, on each round the gambler (agent) selects one among $K \in \mathbb{N}$ available arms (actions) and the environment selects a payoff (reward) vector

[6]Multisets generalise sets by allowing repeated elements. In the context of route choice, this translates into routes with repeated links, which is only possible in the presence of cycles (loops).

over the arms. The rewards are random, and their distributions are unknown to the agent. The agent then needs to decide on which arms to play (and in which order) to maximise the its cumulative reward. The problem can be modelled with transparent feedback, where the entire payoff vector is revealed to the agent (LITTLESTONE; WARMUTH, 1994); or opaque feedback, where the environment only reveals the payoff of the chosen arm (AUER et al., 2002; AWERBUCH; KLEINBERG, 2004). The transparent model has been more extensively investigated in the literature, but the opaque one is much more challenging and better represents traffic settings (given that a driver only perceives the cost of its current route). Despite their similarities, the multi-armed bandit and route choice problems are conceptually different. Whereas in the former the rewards are simply random variables, in the latter they are a function of the choices made by *all* drivers. Such a dependence on what everyone else is doing poses an additional layer of complexity to an agent's decision process, thus making route choice more challenging.

Another variation of route choice is the dynamic shortest paths problem (a.k.a. en-route trip building and link-based route selection) (AWERBUCH; KLEINBERG, 2004; BAZZAN; KLÜGL, 2008; GRUNITZKI; RAMOS; BAZZAN, 2014). As opposed to route choice, the set of routes here is not known a priori by the agents. In this sense, starting at their origin nodes, agents need to explore the entire road network until a route to their destination is found. In general, learning in this kind of scenario takes much longer than when the set of routes is known a priori. Empirically, however, reasonable results have been achieved here. Although interesting from the learning perspective, nonetheless, we emphasise that this problem lacks realism. Assuming that drivers have no previous knowledge at all about their routes is pointless: in reality, most drivers have at least an idea on how to reach their destinations before they start their trips.

2.1.3 Using non-local information in route choice

As discussed in Section 2.1.1, drivers aim at minimising their travel costs. It is worth noting, however, that drivers have *limited knowledge*, meaning that they are not fully aware of the traffic conditions when making their decisions. Notwithstanding, as drivers become experienced, they tend to take better decisions. Nevertheless, considering the dynamic characteristic of traffic, it may take long for an agent to achieve a reasonable knowledge level (i.e., one that permits the agent to take good decisions).

The use of non-local[7] information to overcome the aforementioned limitation has shown promise. In fact, literature has shown that when drivers have a more complete knowledge on their routes, they can better choose among them (HALL, 1996). The rationale behind providing non-local information to the agents is that it can improve the confidence with which drivers choose their routes. For instance, if an agent's route is not used for a long time, then the estimated cost on that route may be outdated; consequently, comparing a frequently used route and a non-frequently one may be misleading. Therefore, the use of non-local information leads the agents to improve their policy faster.

There is a plethora of approaches concerned with providing and employing non-local information into the agents' decision process. We refer the reader to Zhang et al. (2011) and Essen et al. (2016) for a more thorough review. Here we briefly comment on two kinds of approaches based on the information source: from a central authority, and by communicating with other agents.

Some approaches consider that a central authority recommends routes to the drivers, such as Vasserman, Feldman and Hassidim (2015) and Klügl and Bazzan (2004). The idea is that such a mechanism has a more complete overview of the network conditions, thus being able to provide more accurate information to the agents. In general, however, such approaches consider that the central authority has detailed information about the traffic conditions. On the other hand, several works have considered the case where no such central authority exist. Here, agents interact with each other in order to exchange information. By doing so, these works usually drop the assumption of a full-knowledged authority (HASAN et al., 2016). In general, however, such works make impractical assumptions on agents' knowledge. More details on these approaches are presented in Section 4.4.

In this thesis, we consider the use of non-local information as given by a central authority. However, as opposed to other works, our assumptions regarding the authority's knowledge are very limited. In particular, we consider that the authority have *estimates* on the routes travel times, which may be incorrect. Moreover, the received information is not directly used, but encoded into the agents knowledge. The complete details are presented in Chapter 4.

[7] We employ the term *non-local* instead of *global* because the information to which we refer here may come from other agents, who also only have access to local information.

2.1.4 System-efficient equilibria in route choice

Previously, we seen that the self-interested behaviour of drivers leads them to choose actions that minimise their travel costs. Consequently, the system converges to the UE, which is inefficient from the system's performance. Recall that the level of inefficiency in the system's performance due to drivers selfishness is referred as the Price of Anarchy (PoA). In this section, we briefly discuss the literature focused on minimising the PoA by biasing the UE towards the SO.

The SO is only attainable if the agents behave altruistically. Several works follow this line by explicitly assuming that agents present altruistic behaviour, such as Chen and Kempe (2008), Levy and Ben-Elia (2016), and Hoefer and Skopalik (2009). In fact, according to Fehr and Fischbacher (2003), under certain conditions, real drivers are willing to take altruistic decisions so as to improve the global performance. However, the higher the price of such a social behaviour, the less frequent it is. Hence, altruism cannot be imposed on the agents.

The use of route guidance mechanisms to bias drivers' decisions towards the SO has also been approached in the literature, including Lujak, Giordani and Ossowski (2015) and Bazzan and Klügl (2005). A good review on the topic is provided by Essen et al. (2016). In general, these works assume that a centralised mechanism makes such biased suggestions to the drivers. As discussed by Jahn et al. (2005), some drivers are willing to bear the cost of socially desired routes (up to certain limits) if the traffic system suggests them to do so. However, experiments with human subjects evidence the adoption of such mechanisms is low (ESSEN et al., 2016; RIETVELD, 2010). Moreover, such approaches rely on a central authority with full knowledge.

Another particularly relevant way of enforcing system-efficient behaviour is the use of tolls. In fact, the aforementioned approaches can usually be described in terms of tolls. As opposed to other methods, however, tolls can be enforced on the agents. The idea underlying such pricing mechanisms refers to tolling agents for using links so that they are incentivised to take system-efficient decisions. As discussed in Section 2.1, a fundamental approach here is the marginal-cost tolling (MCT), which is able to align the UE to the SO by charging proportionally to the cost they impose on others.

In general, however, most existing tolling schemes charge drivers *a priori*, i.e., before they actually *start* their trips. Ideally, however, tolls should only be charged after their real marginal costs are available, i.e., at the *end* of the trips. A priori tolling is

indeed appealing from the agents' perspective, since such agents know in advance the toll associated with each of their possible actions. Nonetheless, these schemes usually define the prices based on historical congestion levels, meaning that the agents may end up paying a toll that is higher than their actual marginal costs. In particular, since MCT is based on the impact an agent causes on others, one cannot assess such impact before it happens (except if one can predict future drivers' decisions along their trips). Hence, we say that tolling agents a priori is *unfair* as compared to tolling a posteriori. A more concrete discussion on this effect is presented in Example 5.1. Finally, we emphasise that existing tolling schemes usually rely on a full-knowledged central authority, with the ability of computing and charging the tolls.

In this thesis, by contrast, we assume that tolls are charged *a posteriori* and *per route*, which results in a fairer toll scheme (a complete discussion on this topic is presented in Chapter 5). We then present a general toll formulation that can be computed directly by the agents. In this way, we can simplify the infrastructure requirements for deploying the tolling scheme by assuming that each vehicle has a navigation device responsible for charging the toll whenever a trip is finished. This makes the agents' decision process easier since the drivers can better understand the costs they are being charged, as reported by the National Surface Transportation Infrastructure Financing Commission (2009). Traditional tolling schemes could also benefit from connected navigation devices. However, such approaches would strongly depend on stable communication (otherwise tolls would not be available a priori), whereas our approach remains robust even under precarious communication conditions (since tolls could be computed at any time after each trip is finished).

2.2 Reinforcement learning

2.2.1 Fundamentals

Reinforcement learning (RL) is the problem of an agent learning its behaviour by reward and punishment from interactions with its environment (SUTTON; BARTO, 1998). The basic RL cycle can be described as follows. Initially, an RL agent observes the current state of the environment and chooses an action based on its knowledge. Afterwards, the agent executes the chosen action and receives a reward, which is then used to update its knowledge base. An agent's knowledge here refers to its *policy*, i.e., a mapping from states to actions. A complete RL cycle is called an episode.

Typically, the RL problem is described within the framework of Markov decision processes (MDPs). An MDP is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, r)$, where:

- $\mathcal{S}$ is the set of environment states where the agent may be situated in;
- $\mathcal{A}$ is the set of actions that the agent can execute (some actions may be available only in specific states);
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition function that determines the probability $P(s' \mid s, a)$ with which the agent reaches state s' after taking action a in state s;
- $r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is a function that specifies the reward $r(s, a)$ that the agent receives after taking action a in state s.

The objective of an RL agent is to learn a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that maximises its cumulative reward over time. Intuitively, $\pi(a \mid s)$ can be seen as the probability with which the agent takes action a while in state s.

In the context of route choice, the actions of an agent can represent the choice of routes between its origin and destination. We can define the reward[8] received after taking action $a \in \mathcal{A}$ in the same way as in Equation (2.5), i.e., $r(a) = r(R)$, with $a = R$. Recall that route choice represents a commuting scenario with daily trips occurring under approximately the same conditions (as discussed in Section 2.1). We also remark that drivers know their routes a priori (or at least a subset of them) and just need to decide on *which* one to take everyday (as opposed to the dynamic shortest path problem, discussed in Section 2.1.2). In this sense, whenever a driver takes a route, it will inevitably reach its destination, thus rendering the *state* definition irrelevant here. Therefore, this problem is typically modelled as a stateless MDP. Further details on this modelling are presented in the next subsection.

Solving a stateless MDP involves finding a policy π (i.e., which route to take) that maximises the agent's average reward. When the model of the environment dynamics (i.e., the reward function r) is known by the agent, finding such an optimal policy is trivial. However, this is rarely the case. To tackle this limitation, the agent must repeatedly interact with the environment to learn a model of its dynamics. A class of RL algorithms particularly appropriate for this setting comprises the so-called temporal-difference (TD) learning algorithms, through which an agent can learn without an explicit model of the environment.

[8]Observe that, although the reward an agent receives is formulated as a function of its single route, it actually depends on the flow (i.e., the number of vehicles) on the links that comprise that route. This is expressed by means of the VDF function, as explained in Section 2.1.

The Q-learning algorithm is a commonly-used TD-based method (WATKINS; DAYAN, 1992). In the case of a stateless MDP, a Q-learning agent learns the expected return $Q(a)$ of selecting each action a by exploring the environment. Such a process must balance exploration (gain of knowledge) and exploitation (use of knowledge). A typical strategy to this end is known as ϵ-greedy exploration, in which the agent chooses a random action with probability ϵ (exploration) or the best action according to its current knowledge with probability $1 - \epsilon$ (exploitation), with $\epsilon \in (0, 1]$. After taking action a and receiving reward $r(a)$, the stateless Q-learning algorithm updates $Q(a)$ using the Q-learning update rule, as follows:

$$Q(a) = (1 - \alpha)Q(a) + \alpha r(a), \tag{2.7}$$

where the learning rate $\alpha \in (0, 1]$ weights how much of the previous estimate should be retained. Roughly, the Q-learning update rule works by adjusting the expectation over an action's value (i.e., the previous Q-value) towards its actual value (i.e., the received reward r) with an α step size. The Q-learning algorithm is guaranteed to converge to an optimal policy if (i) all state-action pairs are experienced an infinite number of times and (ii) the learning and exploration rates go to zero in the limit (WATKINS; DAYAN, 1992). In this regard, the learning and exploration rates are typically multiplied by decay rates $\lambda \in (0, 1]$ and $\mu \in (0, 1]$, respectively, so that, at time t, $\alpha(t) = \alpha\lambda^t$ and $\epsilon(t) = \epsilon\mu^t$.

2.2.2 Multiagent reinforcement learning

Observe that, up to this point, we considered the traditional single-agent RL setting. When multiple agents share a common environment, their actions may affect the reward received by others. As such, the agents need to adapt to each other. Although such effect may seem unimportant, it invalidates the so-called Markov property, i.e., the environment is no longer stationary (TUYLS; WEISS, 2012). Consequently, under these settings, RL algorithms designed to solve MDPs may not work. We refer the interested reader to Tuyls and Weiss (2012) and Buşoniu, Babuska and Schutter (2008) for insightful reviews on multiagent reinforcement learning. Additionally, considering the connection of this topic with game theory, we also recommend the works of Nowé, Vrancx and Hauwere (2012), Leyton-Brown and Shoham (2008), and Nisan et al. (2007).

Multiagent RL (MARL) problems may be approached as stochastic (or Markov)

games (LITTMAN, 1994). Stochastic games can be represented as a tuple $(\mathcal{P}, \mathcal{S}, \mathcal{A}, \mathcal{T}, r)$. The main difference here to MDPs is that $\mathcal{P}$ represents the set of players (agents), and $\mathcal{A}$ represents the joint action space, i.e., $\mathcal{A} = \mathcal{A}_1 \times ... \times \mathcal{A}_n$, with $\mathcal{A}_i$ representing the actions of agent $i \in \mathcal{P}$. Naturally, the transition and reward functions are now given as a function of the joint actions.

Several algorithms have been proposed for stochastic games. Littman (1994) devised Minimax-Q algorithm, which only applies to zero-sum stochastic games. Hu and Wellman (1998, 2003) introduced Nash-Q, which extends Minimax-Q to general-sum games, but limited to two agents. Later on, Littman (2001) developed Friend-or-Foe-Q (FFQ), which extends previous works to general-sum games. Nonetheless, FFQ only applies to coordination games (i.e., where agents benefit by coordinating their actions) or zero-sum games (i.e., where the reward received by the agents always sum to zero). It should be noted, however, that the route choice problem cannot be solved using these formulations: neither all drivers cooperate (though some indeed can), nor all drivers are better off if just one deviate (SANDHOLM, 2007). Verbeeck et al. (2007) and Vrancx, Verbeeck and Nowé (2010) developed learning automata algorithms for coordination-based games. However, again, traffic cannot be approached as a cooperative problem.

Gradient ascent algorithms were also proposed to handle multiagent learning scenarios. Zinkevich (2003) introduced the Generalised Infinitesimal Gradient Ascent (GIGA) algorithm for two-player two-action general-sum repeated games. Bowling (2005) extended the GIGA algorithm with the "win or learn fast" principle (i.e., a variable learning rate), thus delivering the GIGA-WoLF algorithm, which again only applies to 2-player 2-action games. Abdallah and Lesser (2006) presented the Weighted Policy Learner (WPL) algorithm, which outperform the others but whose convergence guarantees still apply to 2-player games, only.

Considering the above limitations, literature has also considered a simpler modelling where each agent's decision process is modelled as an individual MDP. Consequently, an agent interprets the behaviour of other agents as the dynamics underlying its environment. A particularly representative example of such approach was presented by Claus and Boutilier (1998), and refers to such agents as *independent learners*.

In this thesis, we employ the latter aforementioned case, where each independent Q-learner has its own MDP and ignores the other agents' actions. The route choice problem can then be modelled as a stateless MDP, with actions representing routes and rewards expressed as in Equation (2.5). In spite of its simplicity, this modelling properly repre-

sents real traffic settings. The point is that, when a driver is deciding on which route to take, it does not explicitly consider how others are deciding. In fact, drivers have a very limited knowledge about what others are doing (not to mention their policies). Therefore, the stateless modelling represents a suitable way of approaching route choice.

In terms of convergence, the stateless-based modelling of MARL tends to obtain reasonable empirical results (e.g., Hennes, Kaisers and Tuyls (2010), Proper and Tumer (2013), Ramos and Grunitzki (2015)). However, formal guarantees are not usually provided. In this sense, we advance the state of the art by investigating two situations where such guarantees are attainable in the context of route choice. Firstly, in Chapter 3 we show that, when agents use the regret associated with their actions as reinforcement signal, then they converge to the UE. Secondly, in Chapter 5 we consider the case where MCT (discussed in Sections 2.1.1 and 2.1.4) leads the agents to the SO.

2.3 Regret minimisation

In this section, we present a succinct overview of the regret (minimisation) literature. The interested reader is referred to Blum and Mansour (2007) and Cesa-Bianchi and Lugosi (2006) for a more detailed overview.

The notion of regret has been employed from different perspectives since the 1950s. From the decision theory perspective, the so-called regret theory (RT) was developed independently by Bell (1982), Fishburn (1982), and Loomes and Sugden (1982). RT postulates that agents' decisions are affected not only by the utility associated with them, but also by the anticipated disutility (regret) for *not* taking a better decision. In other words, RT claims that agents try to avoid regret when taking decisions. As a descriptive behavioural model, RT represents a suitable approach to analyse route choice behaviour. In contrast, in this thesis we consider another important perspective of regret, namely that of game theory, which focuses on analysing not only the agents' behaviour, but also how they interact and how their decisions affect each other. This second perspective is the usual focus of regret minimising approaches (CESA-BIANCHI; LUGOSI, 2006; FOSTER; VOHRA, 1999).

The idea of *minimising* regret was introduced in the context of evaluating the performance of learning rules (HANNAN, 1957). The so-called *external regret*[9] of an agent

[9] Alternative regret formulations are also available in the literature. The most adopted formulation, nonetheless, is that of external regret, which we also consider in this thesis. We refer the reader to Blum

measures the difference between its average reward and the reward of the best *fixed action* in hindsight. Precisely, the external regret $\mathcal{R}_i^T$ of agent i up to time T is defined as

$$\mathcal{R}_i^T = \max_{a_i^t \in A_i} \frac{1}{T} \sum_{t=1}^{T} r(a_i^t) - \frac{1}{T} \sum_{t=1}^{T} r(\dot{a}_i^t), \tag{2.8}$$

where $r(a_i^t)$ represents the reward for taking action a_i^t at time t and $\dot{a}_i^t$ denotes the action *actually taken* by agent i at time t. In the context of route choice, recall (see Section 2.1) that the reward of an action (route) corresponds to the sum of its links' rewards. In this sense, considering that routes may overlap, the external regret of an agent is affected not only by its particular decision (route it has taken), but also by all other agents (including those from other OD pairs) whose routes share links with it. The notion of external regret is presented more intuitively in the next example.

Example 2.1. *Consider again the abstract road network presented in Figure 2.1. Suppose a driver wants to travel from n_4 to n_3, which can be performed using either route $R_1 = \{\{n_4, n_2\}, \{n_2, n_3\}\}$ or route $R_2 = \{\{n_4, n_2\}, \{n_2, n_1\}, \{n_1, n_3\}\}$. For simplicity, assume that travel times are fixed, e.g., the cost on link $\{n_4, n_2\}$ is 8 (as shown in the figure) regardless on the number of agents using it. Moreover, let the reward associated with a route be the negative of its travel time. In this sense, the reward on routes R_1 and R_2 are -31 and -25, respectively. Additionally, assume that, as soon as the driver completes its trip, it can magically observe the reward on both routes. If the driver takes route R_1, it receives a reward of -31. Consequently, when it realises that R_2 has reward -25, its regret will be $\mathcal{R} = -25 - (-31) = 6$. If the agent takes route R_2, on the other hand, then it experiences a regret of $\mathcal{R} = -25 - (-25) = 0$. The agent, therefore, has an incentive to always choose route R_2, which has a lower regret and, consequently, a higher reward.*

An algorithm satisfies the *no-regret property* (a.k.a. Hannan's consistency) if it learns a policy for which $\mathcal{R}_i^T \rightarrow 0$ as $T \rightarrow \infty$ (HANNAN, 1957). Along these lines, regret minimisation can be seen as a natural definition of how rational agents behave over time (BLUM; EVEN-DAR; LIGETT, 2010).

We highlight that regret cannot be computed without knowing the cost of all routes along time (essential for computing the $\max$ operator of Equation (2.8)). For the original purpose of simply evaluating an agent's performance, knowing all costs along time is not a restrictive assumption. On the other hand, in order for regret to be computable by agents,

and Mansour (2007) for a more complete discussion on this and alternative regret formulations.

then the process is more tricky. In fact, part of this thesis (Chapter 3) tries to overcome such limitation by keeping estimates of the actions' rewards, which allow the agents to obtain reasonable estimates relying only on their local knowledge.

In the RL context, regret has been mainly employed as a convergence measure (SHOHAM; POWERS; GRENAGER, 2007; BUŞONIU; BABUSKA; SCHUTTER, 2008). One of the first works to employ regret as the reinforcement signal was that of Hart and Mas-Colell (2000). However, their approach was focused on the correlated equilibrium, in which a mediator recommends actions to the agents (AUMANN, 1974). Other approaches considered regret minimisation in more general cases, but assuming a limited number of agents (BOWLING, 2005) or even assuming they know their regret in advance (ZINKEVICH et al., 2008). In this thesis, we focus on employing regret to guide the learning process but taking into account that, by definition, agents cannot compute their regret exactly (given they can only observe the reward of taken routes). Thus, we show how such values can be estimated by the agents.

Chorus and colleagues (2008, 2010) employed regret to improve predictions of travellers' behaviour in the context of discrete choice models. They proposed the random regret minimisation (RRM) model, in which drivers are said to avoid regret when making decisions. Importantly, though, the RRM model (unlike ours) do not take learning into account and assumes that drivers have full knowledge regarding their regrets and/or travel costs distributions. Furthermore, the travel costs are assumed to be fixed, whereas in practice they change steadily as a consequence of the agents' learning/adaptation process.

Regret has been of particular interest of the online optimisation and congestion games literatures. In online optimisation, the problem can be modelled as multi-armed bandits, and the focus is on minimising the regret associated with the arms (AUER et al., 2002). Along these lines, the regret bound has been consistently refined by Dani, Kakade and Hayes (2007), Abernethy, Hazan and Rakhlin (2012), and Agarwal, Dekel and Xiao (2010), to mention a few. However, frequently assuming agents can observe more than simply their own travel times. The congestion games formulation (ROSENTHAL, 1973; ROUGHGARDEN, 2005) can also be used to approach route choice. In fact, important results have been achieved by Blum, Even-Dar and Ligett (2010). Notwithstanding, as discussed in Section 2.1.2, these models represent the drivers as a set of infinitesimally small agents. In contrast, we assume a finite set of players.

Also important, alternative regret formulations have been proposed in the literature. The idea underlying such formulations is to consider particular aspects of the de-

cision process in order to improve the agents' performance. Some of such formulations include *policy regret*, by Arora, Dekel and Tewari (2012), and *counterfactual regret*, by Zinkevich et al. (2008). In this thesis, we also propose an alternative regret formulation called *action regret*. In contrast to other approaches, however, our formulation enables the agents to estimate their regret, thus being useful in their learning process.

In this thesis, we use the idea of regret to guide the RL process towards the UE. This is the focus of Chapters 3 and 4. Precisely, we formulate the regret of actions and use them as reinforcement signal. The underlying idea of using regret in the learning process is that, as a natural definition on how self-interested agents behave overtime, regret can guide them towards their objective, namely minimise their travel times.

3 LEARNING TO MINIMISE REGRET

In this chapter, we introduce a regret-minimising method through which reinforcement learning agents can learn to choose their best routes using regret. In particular, agents use the regret (rather than reward) associated with their actions as reinforcement signal. The objective is to show that, in the limit, such agents converge to an approximate user equilibrium. In this regard, we develop a method for the agents to estimate their regret locally, based on an internal history of observed rewards. Such regret estimates are then used for updating the agents' policies. We provide a theoretical analysis of the system's convergence, showing that our approach minimises the agents' regret in the limit and thus approximates the user equilibrium.

3.1 Motivation and contributions

As discussed in Chapter 2, reinforcement learning (RL) in multiagent settings is challenging because self-interested agents must adapt to each others' decisions. Regret-minimising algorithms have shown promising here. In this chapter, we investigate how agents can learn with performance guarantees by minimising their regret.

We emphasise that, as seen in Section 2.3, an agent cannot compute its real regret (using Equation (2.8)) due to the lack of information regarding its routes rewards. This is due to fact that regret is measured considering (i) the agent's average reward resulting from its sequence of actions and (ii) the average reward following the best fixed action 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 full-knowledged 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 (STONE; VELOSO, 2000), especially in the highly competitive settings considered here.

In this regard, in the context of route choice, we investigate how agents can estimate their regret based exclusively on local information (i.e., the rewards actually observed by them). The idea underlying our approach is that, if agents can *estimate* the

regret associated with *particular* actions, then such information could be used to guide their learning process. After all, minimising regret in route choice can be intuitively seen as choosing the best routes.

The main contributions of this chapter can be enumerated as follows.

- We define the estimated *action regret*, which measures the regret of single actions. In this way, the Q-value of an action can be updated using the corresponding regret (rather than reward) as reinforcement signal. Moreover, we prove that learning with action regret minimises the agent's external regret.
- We introduce a method for agents to estimate their action regret relying only on their experience (i.e., travel time of current route). In this sense, we eliminate the assumption of full information. We show that such estimates converge to the true values in the limit and that they are useful in the learning process.
- We develop an RL algorithmic solution that employs *action regret as the reinforcement signal* for updating the agent's policy. In this way, the agents learn to choose the actions that minimise their external regret.
- We provide theoretical results bounding the system's performance. Specifically, we show that an agent's average external regret is $O\left(\left(\frac{K-1}{TK}\right)\left(\frac{\mu^{T+1}-\mu}{\mu-1}\right)\right)$ after T timesteps, where K is the number of available routes and μ is the decay rate of the exploration parameter. Moreover, we show that the system converges to a ϕ-approximate UE when all agents use our method, where ϕ corresponds to the bound on the agents' regret.

3.2 Learning to choose routes by minimising estimated regret

This section presents our method for the agents to learn to choose their best routes by minimising regret. To be specific, we model the route choice problem as a stateless MDP and represent drivers by means of Q-learning agents. At every episode, each such agent chooses a route from its origin to its destination and observes the travel time on it. Each agent then computes the regret associated with the chosen route and uses such information to update its Q-table. Considering that agents have limited knowledge, we firstly provide a mean for the agents to estimate the regret associated with their actions based on their own experience (Section 3.2.1). Afterwards, we formulate the stateless MDP and the Q-learning algorithm that uses previously calculated regret estimates to

learn which are the best routes (Section 3.2.2). Later, we provide theoretical (Section 3.3) and empirical (Section 3.4) analyses of our method, showing that it minimises regret and converges to the user equilibrium.

3.2.1 Estimating regret

Let $A_i \subseteq A$ denote the set of actions (i.e., routes) available to agent i. At time t, agent i performs a given action[1] $\dot{a}_i^t \in A_i$ and receives a reward of $r(\dot{a}_i^t)$. We represent the history of estimates of agent i as

$$H_i = \{r(a_i^t) \mid a_i^t \in A_i, t \in [1, T]\},$$

with $r(a_i^t)$ denoting the reward experience of driver i for taking action a at time t. Observe that each agent has its own history of estimates. However, recall that an agent cannot observe the reward of action a_i^t on time t except if it has taken such action at that time, i.e., if $a_i^t = \dot{a}_i^t$. In this sense, we assume that the reward of non-taken actions do not change[2], i.e., the expected reward associated with a non-taken action can be approximated by its most recent observation. Let $\tilde{r}(a_i^t)$ represent the most recent reward *estimate* of agent i for taking action a on time t (either the current reward or the last[3] actually experienced one), as given by Equation (3.1). The history of estimates of agent i can then be rewritten as in Equation (3.2).

$$\tilde{r}(a_i^t) = \begin{cases} r(a_i^t) & \text{if } a_i^t = \dot{a}_i^t \\ \tilde{r}(a_i^{t-1}) & \text{otherwise} \end{cases} \tag{3.1}$$

$$H_i = \{\tilde{r}(a_i^t) \mid a_i^t \in A_i, t \in [1, T]\} \tag{3.2}$$

Given the above definitions, we can now formulate the *estimated action regret* of action a for agent i up to time T as in Equation (3.3). The estimated action regret $\tilde{\mathcal{R}}_{i,a}^T$ can be seen as an estimate of the average amount lost by agent i up to time T for taking action a (latter term) rather than the action with the highest estimated reward (former term).

[1] We use $\dot{a}_i^t$ to distinguish the action taken by agent i at time t from any of its other actions a_i^t in the same time.

[2] This is not a restrictive assumption while computing reward estimates. Further ahead, in Theorem 3.4, we show that our reward estimates indeed converge to their true values in the limit.

[3] As initial value, one can consider the minimum possible reward, i.e., the free flow travel times.

Observe that each agent estimates the action regret associated with *each* of its actions, i.e., the estimated action regret of a route varies from an agent to another.

$$\tilde{\mathcal{R}}_{i,a}^{T} = \max_{b_i^t \in A_i} \frac{1}{T} \sum_{t=1}^{T} \tilde{r}(b_i^t) - \frac{1}{T} \sum_{t=1}^{T} \tilde{r}(a_i^t) \tag{3.3}$$

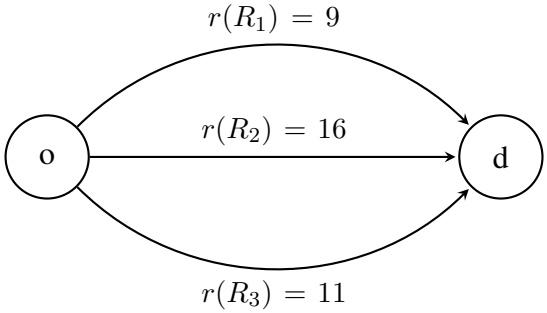
Similarly, we can reformulate Equation (2.8) to obtain the *estimated external regret* of agent i, as in Equation (3.4). The estimated external regret $\tilde{\mathcal{R}}_i^T$ of agent i expresses how much worse it performed, on average, up to time T for not taking only the best fixed action regarding its experience. The main advantage of this formulation over the real regret (Equation (2.8)) is that it can be computed locally by the agents, thus eliminating the need for a central authority. Moreover, as the required information is already available to the agents, they can use such measure to guide their learning process.

$$\tilde{\mathcal{R}}_i^T = \max_{a_i^t \in A_i} \frac{1}{T} \sum_{t=1}^{T} \tilde{r}(a_i^t) - \frac{1}{T} \sum_{t=1}^{T} r(\dot{a}_i^t) \tag{3.4}$$

We highlight that action regret and external regret are fundamentally different despite their similar formulations. Whereas the former considers the reward of a *single, fixed* action, the latter considers reward of the *sequence* of actions actually taken by the agent. An illustrative comparison of these two formulations is presented in Example 3.1. The agents' ultimate objective is to minimise their external regret (or, equivalently, maximise their reward), which is possible by employing action regret as reinforcement signal (further details are discussed in Theorem 3.3 and the accompanying proof). Therefore, action regret represents a mean for the agents to minimise their external regret.

Example 3.1. *Consider the abstract road network presented in Figure 3.1, where a single driver has three possible routes (R1, R2, and R3) for travelling from O to D. For simplicity, assume that the routes' rewards are fixed (though the agent does not know that) and that regret is magically available (someone else—not the agent—computes regret and tells it to the agent). Also assume that, in the first three episodes, for some reason the agent has taken route R1 at the first episode, route R2 at the second episode, and route R3 at the third episode. After this sequence of actions, the average reward received by the agents was* $\frac{9+16+11}{3} = 12$. *Observe that the agent performed poorly as compared to if it has taken only route R2 (which has the highest average reward). Hence, the external regret of the agent at this point is 16-12=4. From this regret, the agent knows that it has performed poorly, but it does not know which actions were responsible for that performance. That is why action regret plays a role here. The action regret of route R1 is computed as the*

Figure 3.1: Illustrative comparison of external regret and action regret on a simple 3-route network. The link' labels represent the reward associated with them.



difference between its average reward and that of the best route (i.e., R2, with an average reward of 16), thus accounting for 6. Similarly, the regret of routes R2 and R3 are 0 and 1, respectively. Therefore, using the action regret, the agent can conclude that choosing route R2 translates into the smallest regret and, thus, highest reward.

At this point, the sharp-eyed reader will wonder why the regret formulation equally weights old and recent rewards. In fact, several alternative formulations are possible here, such as discounting old rewards or even considering a fixed time window, just to mention a few. However, by definition, regret considers the overall performance of the agent (or action) as compared to the best fixed action. Reducing the regret horizon would make the agent ignore previous knowledge, which could lead to local optima. Consequently, the system could get stuck in infinite loops among local optima. Notwithstanding, more sophisticated (and carefully designed) regret formulations could be useful for, e.g., ignoring outdated information. Such a direction was left as future work. Nevertheless, our work provides a building block towards more sophisticated regret formulations.

3.2.2 Learning to minimise estimated regret

Building upon the regret estimations from the previous section, we now present a simple yet effective RL algorithm enabling the agents to learn a no-regret policy. An overview of our method is presented in Algorithm 3.1. The problem is represented as a stateless MDP and each driver $i \in D$ as a Q-learning agent. The set of actions $A_i = \{a_1, \ldots, a_K\}$ available to agent i corresponds to the routes it can use to reach its destination from its origin. The set of all agents' actions is denoted by $\mathcal{A} = \{A_i \mid i \in D\}$. Observe that if two agents i and j have the same OD pair, then $A_i = A_j$ (i.e., they have

the same routes). Following Equation (2.5), the reward that agent i receives for taking route a_i^t at episode (or timestep) t is

$$r(a_i^t) = -C_{a_i^t},$$

where $C_{a_i^t}$ denotes the cost experienced in route a_i^t, (as defined in Equation (2.2)). We remark that, in the present (toll-free) setting, a route's cost is based exclusively on the travel time on it, i.e., using Equation (2.1). The drivers' objective is to maximise their cumulative reward.

The learning process works as follows. At every episode $t \in [1, T]$, each agent $i \in D$ chooses an action $\dot{a}_i^t \in A_i$ using the ϵ-greedy exploration strategy (line 5 of Algorithm 3.1). The exploration rate ϵ at time t is given by $\epsilon(t) = \mu^t$. After taking the chosen action, the agent receives reward $r(\dot{a}_i^t)$. Afterwards, the agent updates its history H_i using Equation (3.1) (lines 8–10 of Algorithm 3.1) and calculates the estimated regret of action $\dot{a}_i^t$ using Equation (3.3) (line 11 of Algorithm 3.1; recall that the action regret can be easily computed as soon as H_i was already updated). Finally, the agent updates $Q(\dot{a}_i^t)$ *using the estimated action regret for that action as reinforcement signal*, as in Equation (3.5) (line 12 of Algorithm 3.1). The learning rate α at time t is given by $\alpha(t) = \lambda^t$.

$$Q(\dot{a}_i^t) = (1 - \alpha)Q(\dot{a}_i^t) + \alpha\tilde{\mathcal{R}}_{i,\dot{a}_i^t}^t \tag{3.5}$$

Recall that the estimated action regret is used as reinforcement signal (i.e., for updating an agent's policy). The Q-table of an agent then encodes an expectation over its actions' regrets. Since the regret of an action is a function of its reward, we have that the lower its regret, the higher its reward. Thus, using regret as reinforcement signal leads to minimising an agent's estimated external regret. A formal proof on this is presented in Theorem 3.3.

We highlight that the original definition of external regret in Equation (2.8) considers the average reward of the agent over all actions it has taken. Specifically, it accounts for actions with both high (exploitation) and low (exploration) rewards. The problem is that the agent cannot identify which actions deteriorate its average reward, thus leading the regret associated with good-performing actions to be penalised by that of bad-performing ones. Moreover, the learning process works by adjusting the expected value (estimated regret) of each action of the agent, which is not possible without knowing the contribution

Algorithm 3.1: Regret-minimising Q-learning (for agent i)

input: set of actions A_i, learning decay rate λ, exploration decay rate μ, number of episodes T

initialise Q-table: $Q(a_i) \leftarrow 0 \; \forall a_i \in A_i$;
initialise history of estimates: $H_i \leftarrow \emptyset$;
for $t \in \{1, \ldots, T\}$ **do**
  $\alpha \leftarrow \lambda^t$; $\epsilon \leftarrow \mu^t$; `// update learning and exploration rates`
  $\dot{a}_i^t \leftarrow \epsilon$-greedy; `// choose (and take) action using` ϵ`-greedy`
  $f_{\dot{a}_i^t} \leftarrow$ observe travel time on $\dot{a}_i^t$;
  $r(\dot{a}_i^t) \leftarrow -f_{\dot{a}_i^t}$; `// compute the reward of` $\dot{a}_i^t$
  for $a_i^t \in A_i$ **do**
    $\tilde{r}(a_i^t) \leftarrow \begin{cases} r(a_i^t) & \text{if } a_i^t = \dot{a}_i^t \\ \tilde{r}(a_i^{t-1}) & \text{otherwise} \end{cases}$; `// update estimate` $\tilde{r}(a_i^t) \in H_i$
  end
  $\tilde{\mathcal{R}}^T_{i,\dot{a}_i^t} \leftarrow \max_{b \in A_i} \frac{1}{T} \sum_{u=1}^{T} \tilde{r}(b_i^u) - \frac{1}{T} \sum_{u=1}^{T} \tilde{r}((\dot{a}_i^t)_i^u)$; `//compute regret of` $\dot{a}_i^t$
  $Q(\dot{a}_i^t) \leftarrow (1-\alpha) Q(\dot{a}_i^t) + \alpha \tilde{\mathcal{R}}^t_{i,\dot{a}_i^t}$; `// update Q-value of` $\dot{a}_i^t$
end

of each action in particular. In other words, the external regret per se is not useful in the learning process. To overcome such limitations, our estimated action regret formulation decomposes the regret per action, i.e., it measures the regret of an action accounting for none but its own rewards. In this sense, an action's regret is not affected by the reward associated with other actions. Using this formulation, an agent can evaluate how much a particular action contributes to its regret. The estimated action regret is therefore *more suitable to evaluate how promising a given action is as compared to the others*. Hence, action regret can be used to guide the learning process.

We remark that Algorithm 3.1 is an abstract scheme of the algorithm from the agent's perspective. In order to effectively run our approach, one needs firstly to load the problem instance and initialise the agents. Afterwards, for every episode (up to episode T), we can run once the main loop of each agent (i.e., where an agent chooses an action, observes the reward, and updates its Q-table). The time and space complexity of our approach are presented in the next proposition. We refer the reader to Appendix A for the proof and for complete details on the simulation procedure.

Proposition 3.1. *Our regret-minimising Q-learning approach has* $O(T(dK + ld + lmK))$ *time complexity and* $O(dK)$ *space complexity, for* T *episodes,* d *drivers, and* K *actions,* l *links, and* m *OD pairs.*

3.3 Theoretical analysis

In this section, we analyse the theoretical aspects of our method. Specifically, our objective is to prove that our method converges to an approximate UE. For simplicity and without loss of generality, we assume that the actions' rewards are in the interval $[0, 1]$.

We begin with the big picture of our analysis. Initially, we show that the environment is stabilising, i.e., randomness is decreasing along time (Theorems 3.1 and 3.2). We then analyse the expected reward and regret of the agents (Proposition 3.3). Afterwards, we define a bound on the algorithm's expected regret (Theorem 3.6). Building upon such a bound, we prove that the algorithm is no-regret and converges to an approximate UE (Theorem 3.7).

As a first step, the next proposition defines the probability that best[4] and non-best actions are chosen by a given agent i at episode t.

Proposition 3.2. *Using ϵ-greedy exploration with $\epsilon(t) = \mu^t$, at episode t agent i chooses its best action $\overset{+}{a}{}_i^t = \arg\max_{a_i^t \in A_i} Q(a_i^t)$ with probability $\rho(\overset{+}{a}{}_i^t) = 1 - \frac{\mu^t(K-1)}{K}$ and any other action $\bar{a}_i^t \in A_i \setminus \overset{+}{a}{}_i^t$ with probability $\rho(\bar{a}_i^t) = \frac{\mu^t(K-1)}{K}$.*

Proof. In a given episode t, by definition, the ϵ-greedy strategy *exploits* the best action $\overset{+}{a}{}_i^t = \arg\max_{a_i^t \in A_i} Q(a_i^t)$ with probability $1 - \epsilon$ or *explores* any action $\bar{a}_i^t \in A_i$ with probability ϵ. Observe that the best action can also be selected under exploration. In this sense, the best action is selected with probability $(1-\epsilon) + \frac{\epsilon}{K}$. A non-best action (i.e., any action except for the best one), on the other hand, is selected with probability $\epsilon - \frac{\epsilon}{K}$. Now, considering that the value of ϵ at episode t is given by μ^t, we can rewrite the probability of agent i selecting the best action at that given episode as follows:

$$\begin{aligned} \rho(\overset{+}{a}{}_i^t) &= (1 - \mu^t) + \tfrac{\mu^t}{K} \\ &= 1 + \tfrac{\mu^t - K\mu^t}{K} \\ &= 1 - \tfrac{\mu^t(K-1)}{K}. \end{aligned} \tag{3.6}$$

Similarly, we can rewrite the probability of agent i selecting *any* non-best action as:

$$\begin{aligned} \rho(\bar{a}_i^t) &= \mu^t - \tfrac{\mu^t}{K} \\ &= \tfrac{K\mu^t - \mu^t}{K} \\ &= \tfrac{\mu^t(K-1)}{K}, \end{aligned} \tag{3.7}$$

which completes the proof. □

[4]Hereafter, we refer to the action with highest Q-value as the *best action* and to the other actions as *non-best*. Observe that the best action is not necessarily optimal.

We can now formulate Theorem 3.1.

Theorem 3.1. *The environment is stabilising.*

Proof. We say the environment is stabilising if randomness is decreasing along time. Observe that such a randomness is the result of agents exploration, i.e., the environment is more stable when exploration is low.

As the agents are using the ϵ-greedy strategy, the exploration is defined in terms of the ϵ parameter. Recall that ϵ is the same for all agents and it depends only on the decay rate μ and current timestep, i.e., the value of ϵ at time t is given by $\epsilon(t) = \mu^t$. From Proposition 3.2, we have that at episode t agent i chooses its best action $\overset{+}{a}{}_i^t = \arg\max_{a_i^t \in A_i} Q(a_i^t)$ with probability $\rho(\overset{+}{a}{}_i^t) = 1 - \frac{\mu^t(K-1)}{K}$ and any other action $\bar{a}_i^t \in A_i \setminus \overset{+}{a}{}_i^t$ with probability $\rho(\bar{a}_i^t) = \frac{\mu^t(K-1)}{K}$. For simplicity, hereinafter we will refer to $\rho(\overset{+}{a}{}_i^t)$ and $\rho(\bar{a}_i^t)$ as $\overset{+}{\rho}{}_i^t$ and $\bar{\rho}_i^t$, respectively, and even omit t and i when they are clear from the context.

We can formulate the change in the best action probability over time as the difference between any consecutive timesteps. Concretely,

$$\begin{aligned}
\Delta\overset{+}{\rho}{}_i^t &= \overset{+}{\rho}{}_i^t - \overset{+}{\rho}{}_i^{t-1} \\
&= 1 - \tfrac{\mu^t(K-1)}{K} - 1 + \tfrac{\mu^{t-1}(K-1)}{K} \\
&= \tfrac{(K-1)(\mu^{t-1}-\mu^t)}{K}.
\end{aligned}$$

Based on Proposition 3.2, observe that as $t \to \infty$ and $\epsilon \to 0$, we have that $\overset{+}{\rho}{}_i^t \to 1$ and $\bar{\rho}_i^t \to 0$, meaning that randomness is decreasing. Moreover, $\Delta\overset{+}{\rho}{}_i^t \to 0$ at the same rate, meaning that the environment is stabilising.

Additionally, observe that the learning rate α may also affect the environment's stability due to abrupt changes in the Q-table. The point is that the Q-value of the true best action may be lowered so that it does not look the best anymore. To avoid this issue, α needs to be low to properly deal with stochastic rewards, some of which may not be representative of the average reward. Similarly to what was assumed for ϵ, α is the same for all agents and it depends only on the decay rate λ and the current timestep t, i.e., the value of α at time t is given by $\alpha(t) = \lambda^t$. Therefore, the maximum change in the Q-values goes to zero as $\alpha \to 0$ and $t \to \infty$. Moreover, the probability of abrupt changes in the best Q-values also goes to zero in the limit (as shown in Theorem 3.2). □

Recall that, although the environment is stabilising, one of the key Q-learning properties is that every action should be infinitely explored. The ϵ parameter ensures this. In fact, the ϵ-greedy exploration strategy does not invalidate the no-regret property,

given that it allows the agents to occasionally explore sub-optimal actions as soon as their average performance is no-regret (BLUM; EVEN-DAR; LIGETT, 2010). It should be noted, however, that even after experimenting every action enough, *abrupt* changes in the Q-values may lead a so far *optimal action to seem sub-optimal*. In fact, even small changes in the Q-values can have this effect. Nonetheless, as the environment is stabilising (Theorem 3.1), the amplitude of such changes needs to be higher to affect the Q-values. Hence, the probability of such abrupt changes goes to zero in the limit. The next theorem demonstrates precisely that. We will refer to such changes as *abrupt* hereinafter.

Theorem 3.2. *Suppose ∇ agents decide to explore a non-best action. The probability that such an event changes abruptly the Q-values of best actions (of any agent) is bounded by $O(\bar{\rho}^{\nabla}(\overset{+}{\rho} + \bar{\rho}))$. Such a probability goes to zero as $t \to \infty$, $\alpha \to 0$ and $\epsilon \to 0$.*

Proof. An abrupt change may occur in the Q-table if the agent receives a reward that leads the Q-value of a non-best action to become better than that of the best one. Recall that, in the case of Q-learning, only the currently taken action has its Q-value updated. In this regard, an abrupt change is only relevant in two cases: (i) the Q-value of the best action drops to below those of other actions, (ii) the Q-value of a non-best action rises to above that of the best action.

Case (i): an abrupt drop of the best Q-value of agent i may occur if it decides to *exploit* its best action $\overset{+}{a}{}_i^t$ while, at the same time, ∇ agents (that so far consider any other action $\overset{+}{a}{}_j^t \neq \overset{+}{a}{}_i^t, \forall j \in \nabla$ as their best one) decide to *explore* their non-best action $\bar{a}_j^t = \overset{+}{a}{}_i^t, \forall j \in \nabla$. Assume that, at this point, agent i receives a reward $r(\overset{+}{a}{}_i^t) > \frac{Q(\bar{a}_i^t) - (1-\alpha)Q(\overset{+}{a}{}_i^t)}{\alpha}$, and that $\nabla > \lceil \frac{Q(\bar{a}_i^t) - (1-\alpha)Q(\overset{+}{a}{}_i^t)}{y\alpha} \rceil$, with $\bar{a}_i^t \in A_i \setminus \overset{+}{a}{}_i^t$ and y representing the contribution of each agent to the reward function (e.g., in Equation (2.4), each agent contributes with -0.02 to the reward). Then, after the Q-value is updated, we have that $\exists \bar{a}_i^t \in A_i \setminus \overset{+}{a}{}_i^t : Q(\bar{a}_i^t) > Q(\overset{+}{a}{}_i^t)$. In the following timestep, the agent shall exploit with probability $\overset{+}{\rho}$ the action $\bar{a}_i^t$ (whose value is $Q(\bar{a}_i^t)$) and the ∇ agents back to their best action, making the reward of $\overset{+}{a}{}_i^t$ once again better than $\bar{a}_i^t$ (indeed, some of them may not back, as the explored action may be better; however, even one agent is enough so that the condition holds). However, at this point, the agent shall exploit with probability $\overset{+}{\rho}$ the action $\bar{a}_i^t$, whose value $Q(\bar{a}_i^t)$ became better than $Q(\overset{+}{a}{}_i^t)$ in the previous step. Therefore, an abrupt rise only occurs if the above scenario happens, whose probability is $\breve{\rho} = \overset{+}{\rho} \times \bar{\rho}^{\nabla}$ and goes to zero as $t \to \infty$.

Case (ii): an abrupt rise of a non-best Q-value of agent i may occur if it decides to *explore* a non-best action $\bar{a}_i^t$ (rather than *exploiting* $\overset{+}{a}{}_i^t$) and ∇ agents from $\bar{a}_i^t$ (that were

exploiting $\bar{a}_i^t$) decide to *explore* any other action. Assuming that, at this point, the agent receives a reward $r(\bar{a}_i^t) > \frac{Q(\overset{+}{a}_i^t)-(1-\alpha)Q(\bar{a}_i^t)}{\alpha}$ and that $\nabla > \lceil \frac{Q(\overset{+}{a}_i^t)-(1-\alpha)Q(\bar{a}_i^t)}{y\alpha} \rceil$, then, after the Q-value is updated, we shall have $Q(\bar{a}_i^t) > Q(\overset{+}{a}_i^t)$. In the following timestep, the ∇ agents back to their best action (again, even one agent is enough), making the reward of $\bar{a}_i^t$ worse than of $\overset{+}{a}_i^t$, and thus leading the agent to believe this action is the best when it actually is not. Therefore, an abrupt rise only occurs if the above scenario happens, whose probability is $\hat{\rho} = \bar{\rho} \times \bar{\rho}^{\nabla} = \bar{\rho}^{\nabla+1}$ and goes to zero as $t \to \infty$.

Putting altogether, we have that the probability of any of the above scenarios is $\check{\rho} + \hat{\rho} = \overset{+}{\rho} \times \bar{\rho}^{\nabla} + \bar{\rho}^{\nabla+1} \leq O(\bar{\rho}^{\nabla}(\overset{+}{\rho} + \bar{\rho}))$, as required. □

The above theorems state that, when the agents are learning, as times goes to infinity, the value of α and ϵ become so small that the probability of noisy observations changing the Q-table (and, mainly, the best action) goes to zero. Observe that an agent can, eventually, change its best action given it *is* learning. However, the agent should be able to prevent its Q-values from reflecting unrealistic observations.

In the long run, we can say that a learning agent explores the available routes until it is confident enough (environment is stable) about the best one (maximising reward). Of course, stability does *not* imply that the Q-value estimates are correct and that the agents are under UE. These are shown later in this section, in Theorems 3.4 and 3.7, respectively.

Having proved that the environment is stabilising, we can turn our attention to the agents' behaviour. Recall that, in our approach, the agents learn using the action regret definition. However, the action and external regret definitions are not equivalent. The next theorem shows that, if an agent employs the action regret in the learning process, then it will minimise its external regret.

Theorem 3.3. *Learning with action regret as reinforcement signal minimises the agent's external regret.*

Proof. Recall that an agent's Q-table provides an expectation over its actions' regret. Specifically, in a certain time t, the action with highest Q-value $\overset{+}{a}_i^t = \arg\max_{a_i^t \in A_i} Q(a_i^t)$ is the one expected to incur agent i with the lowest action regret. Recall that the higher an action's reward, the lower its action regret. Whenever the agent exploits its best action, it receives the highest reward, which decreases its external regret (as shown next, in Lemma 3.1). On the other hand, if the agent decides to explore another action, its external regret increases. However, considering the environment is stabilising and the probability of exploration $\bar{\rho}$ is decreasing, then the agent's external regret approaches zero in the limit.

Therefore, the action that minimises the external regret is precisely the one with smallest action regret. □

Lemma 3.1. *Consider an agent i at timestep t. If the agent exploits its best action (which occurs with probability $\overset{+}{\rho}$), then we have that $\mathcal{R}_i^{T+1} \leq \mathcal{R}_i^T$, i.e., its external regret does not increase.*

Proof. Analysing the external regret formulation, it can only increase if the difference between its terms increases. Considering the environment is stabilising, such change may only occur in the following situations: (i) the agent is exploring, (ii) abrupt changes occur in the Q-values. However, following Theorems 3.1 and 3.2, we have that, in the limit, the probability of the above situations tends to zero. Moreover, even if situation (ii) occurs in the limit, as all actions are infinitely explored, the agent will inevitably update its Q-values so that they reflect the real expectation over its actions. In this case, after the best action is finally found, the agent's external regret stops to increase. □

In Section 3.2.1, we presented a method for estimating the actions' rewards based on the agent's experiences. When *estimating* values, accuracy matters. In our context, a good precision in the reward estimations is fundamental to obtaining good regret estimates. Empirically, we have observed that the higher the precision, the better the agents learn. Thus, establishing bounds on the quality of the action regret estimates is desired.

Theorem 3.4. *The error of any action's estimated reward is $\delta \leq \sqrt{-\frac{\ln(\beta/2)}{2S}}$ in the $(1-\beta)$ confidence interval after the action is sampled S times, with β denoting the probability of the estimation error being at least δ. In other words, after an action is sampled S times, the estimation error is lower than (or equal to) δ with probability greater than (or equal to) $1-\beta$.*

Proof. Here we show that the estimation error tends to zero as time goes to infinity and the environment becomes more stable. Consider an agent i and its set of actions A_i. To analyse the precision of its estimations, we can apply the Hoeffding's bound (HOEFFDING, 1963), which states that:

$$P\left(\left|\tilde{r}(a_i^S) - r(a^S)\right| \geq \delta\right) \leq 2\exp\left(-2S\delta^2\right), \tag{3.8}$$

where S is the number of times agent i has taken action a (i.e., the amount of reward samples for action a), $\tilde{r}(a_i^S) = \frac{1}{S}\sum_{t=1}^{S}\tilde{r}(a_i^t)$ is the average estimated reward, and $r(a^S) = \frac{1}{S}\sum_{t=1}^{S} r(a^t)$ is the true average reward. Let β denote the left-hand side $P(\cdot)$ of the above

inequality. The intuition behind Hoeffding's bound is that, after action a is sampled S times, agent i's estimation on a is no worse than δ with a high probability $1-\beta$. Hoeffding's bound assumes the samples are independent and identically distributed, which is usually not the case, given such samples depend on what other agents are doing. However, given the environment is stabilising and that agents typically have low α (Theorem 3.1), we have that, locally in time, the environment is quasi-stationary. In other words, within any short period of time, actions have similar rewards, meaning they are sampled independently from approximately the same distribution.

Solving Equation (3.8) for S, the minimum amount of samples required for the estimation errors being lower than δ with probability $1-\beta$ is given by Equation (3.9).

$$S \geq -\frac{\ln(\beta/2)}{2\delta^2} \tag{3.9}$$

Moreover, solving Equation (3.8) for δ yields the estimation error in the $(1-\beta)$ confidence interval after S samples, as in Equation (3.10).

$$\delta \leq \sqrt{-\frac{\ln(\beta/2)}{2S}} \tag{3.10}$$

To prove this theorem, one needs to show that the agent chooses each action at least S times so that the above bound holds. We highlight that, in the limit, all actions are chosen infinitely. What remains is to estimate *when* each action will be sampled for the S-th time. In the case of the best action, we have:

$$\begin{aligned}
\textstyle\sum_{t=1}^{T} \overset{+}{\rho} &\geq S \\
\textstyle\sum_{t=1}^{T} \left(1 - \frac{\mu^t(K-1)}{K}\right) &\geq S \\
T &\geq S + \tfrac{\mu(K-1)(\mu^T-1)}{K(\mu-1)} \\
T &\geq S - \tfrac{\mu(K-1)}{K(\mu-1)},
\end{aligned}$$

considering $\mu^T \to 0$ as $T \to \infty$, and for each non-best action we have:

$$\begin{aligned}
\textstyle\sum_{t=1}^{T} \bar{\rho}\left(\frac{1}{K-1}\right) &\geq S \\
\textstyle\sum_{t=1}^{T} \frac{\mu^t}{K} &\geq S \\
T &\geq \tfrac{\log(SK(\mu-1)+\mu)}{\log \mu} - 1.
\end{aligned}$$

From these inequalities, we conclude that every action is sampled enough in the limit, thus completing the proof. □

Corollary 3.1. *Following Theorem 3.4, if we want the estimation error of a given action to be up to* 0.05 *with* 95% *confidence level then, from Equation* (3.9)*, we would need approximately* 738 *samples of that action.*

Observe that the above bound is not tight, given that non-best actions only achieve S samples asymptotically. A further step, left as future work, would be extending the analysis by Auer, Cesa-Bianchi and Fischer (2002). Specifically, their third theorem could be used by defining $\mu = \sqrt[t]{\frac{cK}{d^2 t}}$ and setting $c = d = 1$, thus achieving stronger results.

We now provide a bound on the external regret of the agents, which is useful for establishing the bound on the UE. We begin with the following proposition, which defines the expected instantaneous reward and regret of the agents. We call these values *instantaneous* because they refer to a single timestep (rather than the average over all timesteps) and *expected* to account for the stochastic nature of the choices.

Proposition 3.3. *The expected instantaneous reward* $\mathbb{E}[r_i^t]$ *of agent* i *at time* t *is*

$$\mathbb{E}[r_i^t] = \left(1 - \frac{\mu^t(K-1)}{K}\right) r(\overset{+}{a}{}_i^t) + \frac{\mu^t}{K} \sum_{\bar{a}_i^t \in A_i \setminus \overset{+}{a}{}_i^t} r(\bar{a}_i^t),$$

and the expected instantaneous regret $\mathbb{E}[\mathcal{R}_i^t]$ *of agent* i *at time* t *is given by*

$$\mathbb{E}[\mathcal{R}_i^t] = r(\overset{+}{a}{}_i^t) - \mathbb{E}[r_i^t].$$

Observe that $\mathbb{E}[r_i^t] \to r(\overset{+}{a}{}_i^t)$ *as* $\epsilon \to 0$ *and* $t \to \infty$*. Moreover,* $\mathbb{E}[\mathcal{R}_i^t] \to 0$ *as* $\mathbb{E}[r_i^t] \to r(\overset{+}{a}{}_i^t)$.

The above proposition holds no matter whether the environment is stabilising or not, given the instantaneous regret measures only the difference to the best action at that specific time t. This proposition would not hold only if the best action changes, which occurs with a small probability, as shown in Theorem 3.1. However, recall that we work with estimates over the actions' rewards. This point is discussed in the next theorem.

Theorem 3.5. *Let* $\overset{*}{b}{}_i^t = \arg\max_{a_i^t \in A_i} r(a_i^t)$ *be the action with highest (true) reward and* $\tilde{b}_i^t = \arg\max_{a_i^t \in A_i} \tilde{r}(a_i^t)$ *be the action with highest estimated reward at time* t *for agent* i*. If* $\max_{a_i^t \in A_i} \tilde{r}(a_i^t) \approx \max_{a_i^t \in A_i} r(a_i^t)$ *as* $t \to \infty$*, then* $\tilde{b}_i^t = \overset{*}{b}{}_i^t$ *with high probability. Thus, the instantaneous regret of agent* i *at time* t *is* 0 *with probability* $\left(1 - \frac{\mu^t(K-1)}{K}\right)$*, which approaches* 1 *as* $t \to \infty$*.*

Proof. The agent selects its best estimated action (that with highest Q-value) with probability $\overset{+}{\rho}$. Regret is measured considering the agent's expectation over received rewards. So, according to its current Q-values, selecting the best estimated action yields regret zero.

Observe that having good accuracy is not enough for ensuring that the best estimated action is indeed the best one. However, from Theorem 3.4, it follows that, in the limit, $\tilde{r}(a_i^t) \approx r(a_i^t)$ with probability $(1-\beta)$ for every action $a \in A$. Moreover, recall that such a probability goes to 1 as $t \to \infty$.

Therefore, whenever the agent selects its best estimated action, its instantaneous regret will be zero plus an estimation error δ with probability $(1-\beta)$. Such expected instantaneous regret can be formalised as:

$$\begin{aligned}\mathbb{E}[\mathcal{R}_i^t] &= r(\overset{+}{a}_i^t) + \delta - (\mathbb{E}[r_i^t] + \delta) \\ &= r(\overset{+}{a}_i^t) - \mathbb{E}[r_i^t] + 2\delta.\end{aligned}$$

Finally, observe that $\overset{+}{\rho} \to 1$ and $\delta \to 0$ as $t \to \infty$. Consequently, $\mathbb{E}[\mathcal{R}_i^t] \to 0$. □

We now analyse the regret of our approach.

Theorem 3.6. *The regret achieved by our approach up to time T is bounded by*

$$O\left(\left(\frac{K-1}{TK}\right)\left(\frac{\mu^{T+1}-\mu}{\mu-1}\right)\right).$$

Proof. To establish an upper bound on the regret of any agent i, we need to consider the worst case scenario. Assume that there exists a single optimal action $\overset{*}{a}_i$ with (true) reward always 1 and that every other (sub-optimal) action $\bar{a}_i \in A_i$ has reward 0. In such scenario, we can ignore the estimation error because the rewards are bounded in the interval $[0, 1]$. In the worst case, the agent always chooses a sub-optimal action, which yields an instantaneous regret of 1. However, recall that the agents tend to exploit their best actions. Regret, then, needs to be analysed in expectation.

By employing Proposition 3.3, we can formulate the accumulated expected reward $\mathbb{E}[r_i^T]$ of agent i up to time T as

$$\begin{aligned}\mathbb{E}[r_i^T] &= \tfrac{1}{T}\sum_{t=1}^{T}\mathbb{E}[r_i^t] \\ &= \tfrac{1}{T}\sum_{t=1}^{T}\left[\left(1-\tfrac{\mu^t(K-1)}{K}\right)r(\overset{*}{a}_i^t) + \tfrac{\mu^t}{K}\sum_{\bar{a}_i^t \in A_i \setminus \overset{*}{a}_i^t} r(\bar{a}_i^t)\right] \\ &\leq \tfrac{r(\overset{*}{a}_i^t)}{T}\sum_{t=1}^{T}\left(1-\tfrac{\mu^t(K-1)}{K}\right) + \tfrac{r(\bar{a}_i^t)}{T}\sum_{t=1}^{T}\left(\tfrac{\mu^t(K-1)}{K}\right) \\ &= \tfrac{r(\overset{*}{a}_i^t)}{T}\left(T - \left(\tfrac{K-1}{K}\right)\left(\tfrac{\mu^{T+1}-\mu}{\mu-1}\right)\right) + \tfrac{r(\bar{a}_i^t)}{T}\left(\tfrac{K-1}{K}\right)\left(\tfrac{\mu^{T+1}-\mu}{\mu-1}\right) \\ &= r(\overset{*}{a}_i^t) - \tfrac{r(\overset{*}{a}_i^t)}{T}\left(\tfrac{K-1}{K}\right)\left(\tfrac{\mu^{T+1}-\mu}{\mu-1}\right) + \tfrac{r(\bar{a}_i^t)}{T}\left(\tfrac{K-1}{K}\right)\left(\tfrac{\mu^{T+1}-\mu}{\mu-1}\right) \\ &= r(\overset{*}{a}_i^t) + \left(\tfrac{1}{T}\right)\left(\tfrac{K-1}{K}\right)\left(\tfrac{\mu^{T+1}-\mu}{\mu-1}\right)(r(\bar{a}_i^t) - r(\overset{*}{a}_i^t)).\end{aligned}$$

The third step of the above equation is a consequence of the worst-case assumption that all sub-optimal actions have zero reward. Using the above formulation, we can redefine agent's i external regret up to time T as

$$\begin{aligned}\mathbb{E}[\mathcal{R}_i^T] &= \max_{a\in A_i} \tfrac{1}{T}\sum_{t=1}^{T} r(a) - \tfrac{1}{T}\sum_{t=1}^{T} r(\dot{a}_i^t) \\ &\leq \tfrac{1}{T}\sum_{t=1}^{T} r(\overset{*}{a}_i) - \mathbb{E}[r_i^T] \\ &= r(\overset{*}{a}_i) - \mathbb{E}[r_i^T] \\ &= \left(\tfrac{1}{T}\right)\left(\tfrac{K-1}{K}\right)\left(\tfrac{\mu^{T+1}-\mu}{\mu-1}\right)(r(\overset{*}{a}_i^t) - r(\bar{a}_i^t)).\end{aligned}$$

Again, the second step is a consequence of the worst-case assumption. Observe that $(r(\overset{*}{a}_i^t) - r(\bar{a}_i^t)) \in O(1)$. Therefore, the expected regret of any agent i up to time T is $O\left(\left(\frac{K-1}{TK}\right)\left(\frac{\mu^{T+1}-\mu}{\mu-1}\right)\right)$. Furthermore, since this expression goes to zero as time increases, our algorithm is no-regret. □

We now turn our attention to the final step of our proofs. As an intermediate step, we observe that, under UE, all agents have zero regret.

Proposition 3.4. *Under UE, all agents have zero regret.*

Proof. Under UE, every agent uses its lowest cost route and no other available route has a lower cost. Otherwise, the agent would deviate to such lower cost route. In such case, as the difference between the current and best routes is always zero for all agents, we have that the regret is also zero. Therefore, any set of strategies that reach the UE is no-regret. □

We remark that pure UE not always exist in route choice (ROUGHGARDEN, 2005). A more realistic objective then is to find an approximate UE, as in Definition 3.1. Particularly, we show in Theorem 3.7 that the system converges to a ϕ-UE in that, on average, no driver can increase its reward by more than ϕ after changing its route.

Definition 3.1 (ϕ-UE)**.** *The average cost on all routes actually being used by the agents is within ϕ of the minimum cost route, i.e., no driver has more than ϕ incentive to deviate from the route it has learned.*

Theorem 3.7. *The algorithm converges to a ϕ-UE, where ϕ is the regret bound of the algorithm.*

Proof. The key point to establish a convergence guarantee is to show that, in the limit, the action with the highest Q-value is indeed the optimal one.

From Theorems 3.1 and 3.2, we have that the environment is stabilising and that noisy rewards do not influence the Q-values in the limit. At this point, the agent may have learned the optimal action or not. The latter case would only be possible if the agent were not able to explore every action enough. However, recall that our learning and exploration rates ensure that every action is infinitely explored. In the limit, exploration ensures that the Q-value of the optimal action becomes the highest one. On the other hand, if the optimal action is already learned, then Theorem 3.2 ensures that, in the limit, it will remain with the highest Q-value with high probability. Observe that, even in the unlikely event of an abrupt change in the Q-values, the exploration ensures that the optimal action will eventually become the one with the highest Q-value. Thus, the highest Q-value is that of the optimal action.

Regarding the learning process, recall that the agent takes the action with smallest action regret with higher probability. Given that the agent finds the optimal action in the limit, then such action yields the smallest action regret. Consequently, from Theorem 3.3, the agent will minimise its external regret.

Observe that the external regret considers the average reward of the actions. To this respect, as shown in Lemma 3.1 and considering the environment is stabilising, whenever the agent is exploiting its action with highest Q-value, then its external regret will decrease. Moreover, considering the regret is bounded by $\phi = O\left(\left(\frac{K-1}{TK}\right)\left(\frac{\mu^{T+1}-\mu}{\mu-1}\right)\right)$ (from Theorem 3.6), which goes to zero in the limit, we have that algorithm is no-regret. We highlight that the estimation error of the rewards does not invalidate the no-regret property, as $\delta \to 0$ in the limit.

Finally, considering that the algorithm is no-regret, observe that no driver has more than ϕ incentive to deviate from its optimal action. As the environment is stable in the limit, then such condition approximates the UE condition. An exception would be if the agent discovers that a so far sub-optimal action became the optimal one (i.e., due to some change in the environment). However, as the environment is stabilising, the Q-value of that action will inevitably become the highest one in the limit, and the exploitation thereafter will decrease the agent's regret (from Lemma 3.1). Therefore, the agents converge to a ϕ-UE, which completes the proof. □

3.4 Experimental evaluation

In this section, we empirically analyse the performance of our method. The hypothesis we want to validate is that the use of action regret as reinforcement signal leads reinforcement learning agents to converge to the user equilibrium. Before going into the experiments and results, recall that *learning* means finding the best route to take, which becomes a moving target when the environment is shared by many agents. In this context, *convergence* refers to the point at which the agents keep *exploiting* their knowledge most of the time and the system is *stable* (i.e., agents observe only small fluctuations in their costs). What we show is that, using our approach, such a stable point is close to the UE.

3.4.1 Methodology

In order to empirically validate our theoretical results, we simulate our approach as described in Appendix A. Our approach is tested in the following road networks available in the literature[5]. A summary on these networks is presented in Table 3.1. The most representative such networks are illustrated in Appendix B.

Braess graphs: these are expanded versions (ROUGHGARDEN, 2006; STEFANELLO; BAZZAN, 2016; STEFANELLO; SILVA; BAZZAN, 2016) of the network introduced to explain the Braess (1968)'s paradox. Each such graph is denoted by B^p, with $p \in \mathbb{N}^*$, where B^1 is equivalent to the original graph. The B^p graph has $|N| = 2p + 2$ nodes, $|L| = 4p + 1$ links, and a single OD pair. As Stefanello and Bazzan (2016), we use the $p \in \{1, \ldots, 7\}$ Braess graphs, and define a demand of $d = 4{,}200$ drivers.

Bi-commodity Braess graphs: these are additional expansions of the Braess graphs, but containing two OD pairs instead of one (LIN et al., 2005; STEFANELLO; BAZZAN, 2016). We refer to each such graph as BB^p, with $p \in \mathbb{N}^*$, $|N| = 2p + 6$ nodes, and $|L| = 4p + 4$ links. Following Stefanello and Bazzan (2016), here we employ only the odd instances $p \in \{1, 3, 5, 7\}$ of the graphs and set a demand of $d = 4{,}200$ drivers.

OW: this is a synthetic network introduced by Ortúzar and Willumsen (2011, example 10.1). It comprises $|N| = 13$ nodes, $|L| = 48$ links, 4 OD pairs, and $d = 1{,}700$

[5]The road networks are available at <https://github.com/maslab-ufrgs/network-files>.

Table 3.1: Characteristics of the networks used for validation of our approach.

Network	Nodes	Links	OD pairs	Number of drivers	*avg-tt*[a] under UE
B^1	4	5	1	4,200	20.00
B^2	6	9	1	4,200	30.00
B^3	8	13	1	4,200	40.00
B^4	10	17	1	4,200	50.00
B^5	12	21	1	4,200	60.00
B^6	14	25	1	4,200	70.00
B^7	16	29	1	4,200	80.00
BB^1	8	8	2	4,200	10.00
BB^3	12	16	2	4,200	22.00
BB^5	16	24	2	4,200	50.30
BB^7	20	32	2	4,200	≈123.84
OW	13	48	4	1,700	≈67.16
SF	24	76	528	360,600	20.76

[a] Values reported in the literature (STEFANELLO; SILVA; BAZZAN, 2016; STEFANELLO; BAZZAN, 2016).

drivers. The main challenge here is that the OD pairs have overlapping routes.

SF: an abstract representation of the Sioux Falls city, USA (LEBLANC; MORLOK; PIERSKALLA, 1975). It has $|N| = 24$ nodes, $|L| = 76$ links, 528 OD pairs, and $d =$ 360,600 drivers. This network is widely used in the literature because it is realistic and presents all challenges of the other networks.

The number of possible routes in the above networks can be high. Following the literature, we limit the number of available routes to the K shortest ones[6]. The set of K shortest ones of each OD pair was computed using the KSP algorithm (YEN, 1971). The best value for K (i.e., the one which produces the best results) varies among the different networks, depending on their characteristics.

An experiment corresponds to a complete execution, with T =1,000 episodes (except for the SF network, in which case we set T =10,000 episodes), of our method on a single network. After an execution is completed, we measure (considering the last episode) its performance by means of the average travel time (*avg-tt* hereafter, measured in minutes), the average external regret, and the average proximity to the UE. The latter is formulated as

$$\text{proximity}(x, x^*) = 1 - \frac{|x^* - x|}{x^*}, \tag{3.11}$$

and refers to how close the average travel time (say, x) obtained by the agents is to that of

[6] In the case of the BB networks, among the shortest routes we also included the one with the least number of links, otherwise the UE would not be possible (STEFANELLO; BAZZAN, 2016). This is a particularity of the BB networks given their large number of possible routes.

the UE (say, x^*; as reported in Table 3.1); the closer the value is to 1.0, the better.

We tested different value combinations for the Q-learning's parameters. We ran 30 repetitions for each combination of values for these parameters. The complete tuning process is described in Section 3.4.2. The best results were selected for further analyses in Section 3.4.3. Our results are compared against standard Q-learning (*stdQL*, hereinafter), which uses reward (rather than action regret) as reinforcement signal. In what follows, any claim about whether one approach is better than the other is supported by Student's t-tests at the 5% significance level, except if otherwise stated.

The algorithms, data analysis and plots were all implemented in Python 2.7.

3.4.2 Parameter tuning

In this section, we present the experiments conducted to tune the parameters of Q-learning. We tune three parameters: K (number of routes), λ (decay rate of α) and μ (decay rate of ϵ). As for the number of routes, we used $K \in \{4, 8, 12, 16\}$. We empirically found that intermediate values for K pose no significant difference on the results. As for the decay rates, we used the values $\{0.98, 0.99, 0.995, 0.999\}$, with $\lambda = \mu$. Recall that the learning and exploration rates, α and ϵ, are initialised with 1.0 and multiplied by their decay rates after each episode. Thus, lower decay rates were not used to ensure the agents keep learning/exploring for a longer time. The exception was the SF network, for which we defined $\lambda \in \{0.9995, 0.9997, 0.9999\}$ and $\mu \in \{0.995, 0.997, 0.999\}$ to account for its higher demand (i.e., the number of vehicles on it is two orders of magnitude higher than in the other networks), which makes the optimisation process more complex. In this sense, the values used for λ were higher than for μ to ensure that agents keep learning even after exploration is decreased. We ran 30 repetitions for each combination of values for these parameters and compared such combinations using Student's t-tests at the 5% significance level, except if otherwise stated.

Table 3.2 presents the best-performing combinations of parameters. As seen, larger networks tend to depend on a higher number of routes (i.e., a higher value for K). The rationale here is that the number of possible routes increases with the size of the network. Consequently, a higher value for K is necessary to efficiently spread the traffic on these networks. We can observe the same trend in the case of the decay rates. Recall that the learning and exploration rates, α and ϵ, are multiplied by their decay rates after each episode. In this sense, higher decay rates ensure that the agents keep learn-

Table 3.2: Parameters' configuration that produced the best results for each network.

Network	K	λ	μ
B^1	3	0.99	0.99
B^2	4	0.995	0.995
B^3	4	0.99	0.99
B^4	4	0.99	0.99
B^5	8	0.995	0.995
B^6	8	0.995	0.995
B^7	8	0.995	0.995
BB^1	4	0.98	0.98
BB^3	4	0.995	0.995
BB^5	4	0.995	0.995
BB^7	4	0.995	0.995
OW	8	0.995	0.995
SF	4	0.9999	0.998

ing/exploring for a longer time. Hence, in larger networks (especially when demand is higher), the values of λ and μ need to be higher (which translates into slower decays) to ensure that agents explore their routes sufficiently. Such a relationship is illustrated in Table 3.2 by the Braess graphs, where the parameters increase progressively with the size of the network graphs. Of course, small fluctuations are possible here. An example is the B^2, for which the best decay rates are higher than for B^3. Nevertheless, in this particular network, the 0.99 and 0.995 decay rates achieved almost the same results. Also observe the case of the SF network, where the larger number of vehicles makes the optimisation problem harder than in the other networks. In this case, higher decay rates are necessary.

The results of the above configurations were selected for further analyses in the next subsection.

3.4.3 Results

Table 3.3 presents the performance of our approach in terms of external regret and proximity to the UE in all tested networks. In the table, results represent averages over 30 repetitions, with standard deviations shown in parentheses. The values of the algorithms' parameters are those listed in Table 3.2.

As seen in Table 3.3, our approach outperformed standard Q-learning on average, producing solutions with lower regret and that are closer to the UE. To be specific, our approach decreased the average external regret by 21.5% as compared to standard Q-learning. We emphasise that, although improving regret does not necessarily trans-

Table 3.3: Average performance (with standard deviation in parentheses) of our approach (Ours) and of standard Q-learning (StdQL) on different networks in terms of proximity to the UE and external regret.

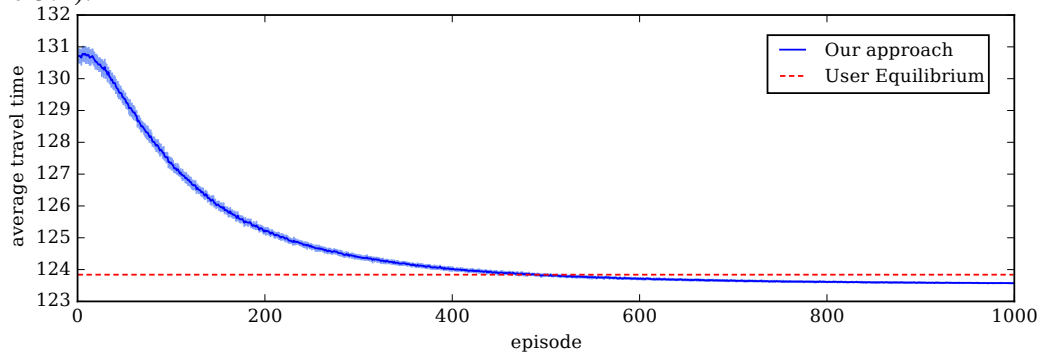
	Proximity to the UE		**External regret**	
Network	**Ours**	**StdQL**	**Ours**	**StdQL**
B^1	1.0000 (0.000)	0.9250 (10^{-2})	0.0057 (10^{-5})	0.0121 (10^{-3})
B^2	0.9981 (10^{-4})	0.9570 (10^{-2})	0.0034 (10^{-5})	0.0111 (10^{-3})
B^3	0.9999 (10^{-5})	0.9974 (10^{-3})	0.0020 (10^{-4})	0.0041 (10^{-3})
B^4	0.9999 (10^{-6})	0.9999 (10^{-6})	0.0004 (10^{-4})	0.0010 (10^{-4})
B^5	0.9916 (10^{-4})	0.9889 (10^{-3})	0.0025 (10^{-4})	0.0039 (10^{-4})
B^6	0.9994 (10^{-4})	0.9996 (10^{-4})	0.0020 (10^{-5})	0.0030 (10^{-5})
B^7	0.9999 (10^{-4})	0.9999 (10^{-5})	0.0025 (10^{-4})	0.0019 (10^{-5})
BB^1	1.0000 (0.000)	1.0000 (0.000)	0.0016 (10^{-5})	0.0016 (10^{-5})
BB^3	0.9991 (10^{-4})	0.9941 (10^{-3})	0.0173 (10^{-5})	0.0178 (10^{-4})
BB^5	0.9991 (10^{-4})	0.9959 (10^{-3})	0.0066 (10^{-5})	0.0063 (10^{-5})
BB^7	0.9979 (10^{-4})	0.9986 (10^{-4})	0.0035 (10^{-5})	0.0029 (10^{-5})
OW	0.9997 (10^{-4})	0.9989 (10^{-4})	0.0161 (10^{-4})	0.0131 (10^{-5})
SF	0.9344 (10^{-4})	0.9887 (10^{-4})	1×10^{-6} (0.0)	2×10^{-6} (0.0)
Average	0.9938 (10^{-4})	0.9880 (10^{-3})	0.0049 (10^{-4})	0.0061 (10^{-4})

lates into better system's performance, it leads to stronger results, given that it better fits the no-regret property. In other words, decreasing regret means increasing convergence probability. In terms of the (absolute) proximity to UE, our regret minimising approach improved upon standard Q-learning by 0.55% on average. Although such an improvement may seem insignificant, it translates into an average *decrease* in the *relative distance* to the UE of 48% as compared to standard Q-learning, meaning that the gap to the UE was cut by a half.

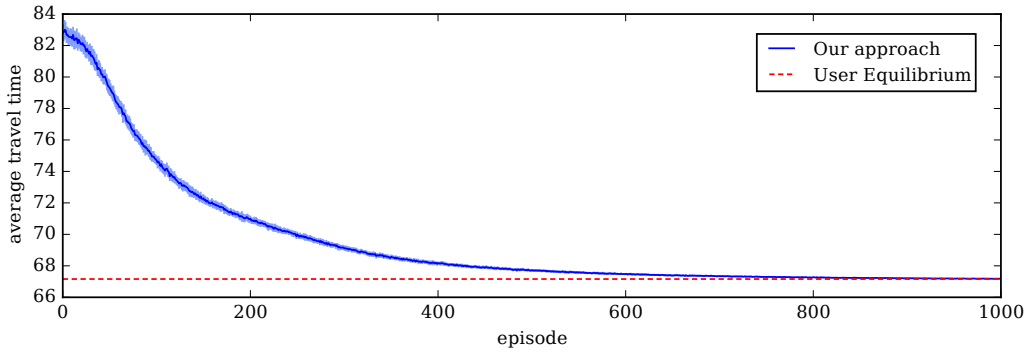
Another relevant aspect to consider here refers to the different parameter values used in the networks. As discussed in Section 3.4.2, the larger the network, the higher the values required for the parameters. We draw attention to the case of the SF network, whose demand is considerably higher than in other networks. This characteristic makes the optimisation problem much harder than in the other networks, which reflected in the comparably worse results achieved for that network. In large scenarios like this, the number of episodes should be increased as well. In fact, we empirically seen that a 5% improvement is easily obtained by increasing the number of episodes. As a proof of concept, however, our results achieved the desired outcome, evidencing that our approach converges to an approximate UE in the limit.

In order to understand how well agents learn when using our algorithm, we can analyse how the average travel time varies along time. Figure 3.2 presents the average

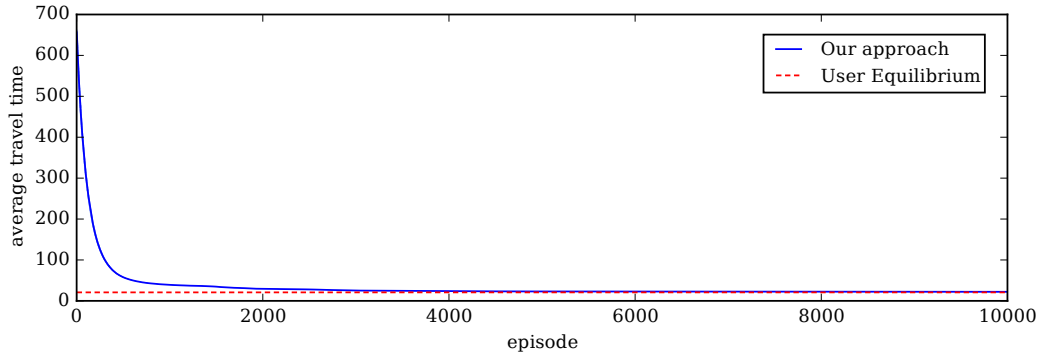
Figure 3.2: Average travel time along episodes in selected networks, with shaded lines representing the standard deviation and dashed lines representing the UE (as reported in Table 3.1).



(a) BB^7 network



(b) OW network



(c) SF network

travel time along episodes in selected networks. In the plots, each curve is an average over 30 repetitions (with standard deviation shown as shaded lines) and dashed lines present the average travel times under UE (as reported in Table 3.1). We emphasise that the apparent faster convergence seen in Figure 3.2 is actually due to the higher number of episodes used for the SF network, which leads to a concentration of the curve in the left side of the plot.

As seen in the plots, *avg-tt* is high in the beginning of the learning process. This is an effect of the Q-table initialisation. As Q-values are initialised with zero, all routes are equiprobable in the beginning. However, as agents explore different routes, the Q-values

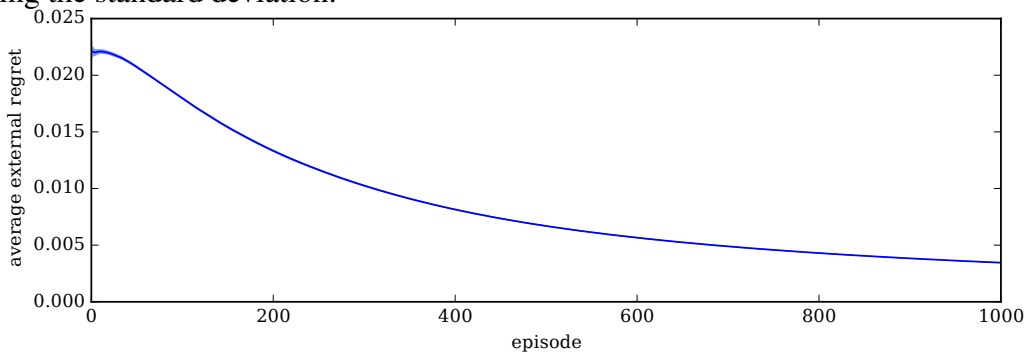
of such routes tend to represent their real costs progressively better, thus allowing the agents to identify which routes are the best. Consequently, *avg-tt* decreases with episodes and converges to an approximate UE. Observe that our algorithm seems to converge to a lower value than that of the UE for the BB^7 network (Figure 3.2a). The reason is that the UE value reported in the literature for this network was computed using CPLEX, with flows represented as *real* numbers. In our approach, however, the flows are *integers* (i.e., each agent controls a unit flow). Consequently, the UE in these two models may differ. In spite of that, the values obtained by our approach are very similar to that of CPLEX in all tested networks.

Convergence speed is strongly related with the network characteristics. One of the key aspects here is the number of OD pairs. In general, the higher the number of OD pairs, the longer it takes for the system to converge. The reason is that as the number of OD pairs increases, the number of overlapping routes also increases. Consequently, agents experiment a higher variability in their route costs estimates. The exploration rate also affects convergence rate. Specifically, the higher the decay rates, the slower agents learn. In reinforcement learning settings, agents must exploit their knowledge as they become experienced. However, given that the environment may change, RL agents cannot completely stop exploring, otherwise they may get stuck with sub-optimal actions. In this regard, the decay rate employed here ensures that the exploration rate is never zero. This is especially important in more complex networks, such as the SF, where agents take longer to learn their routes. In such scenarios, agents need to explore longer before reasonable results are achieved, as evidenced in Figure 3.2c.

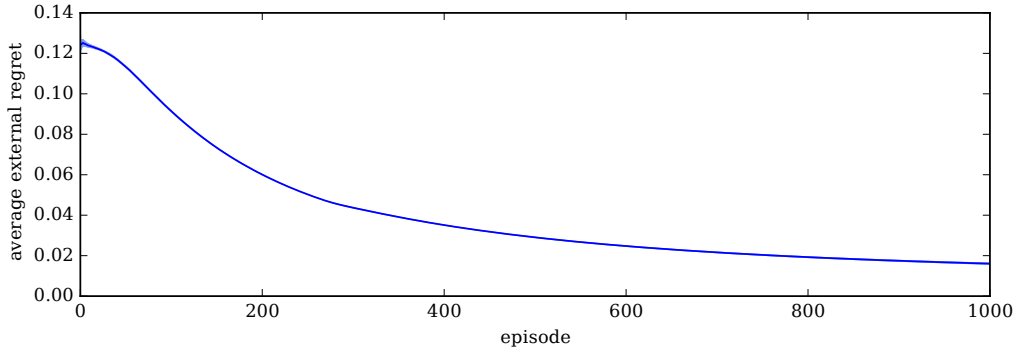
Regarding regret, Figure 3.3 presents the average external regret over time in selected networks. The plots present averages over 30 repetitions, with standard deviation shown as shaded lines. We highlight that the non-smooth behaviour of regret in Figure 3.3c is due to the limited numeric precision used by Python.

From Figure 3.3, one can observe that agents' regret goes to zero as time increases. In fact, that is why the system approaches the UE, given that under UE all agents have zero regret, as seen in Theorem 3.4. Note that regret decreases slightly faster in the OW network than in BB^7, even considering that the former has more OD pairs. The point here is that the BB^7 network has a higher number of agents, which leads to a higher variability in the route cost estimates. Considering that regret measures how much reward an agent loses for not taking the best action, we conclude that regret is sensitive to such variations in route costs. On the other hand, recall that the μ decay rate ensures that exploration

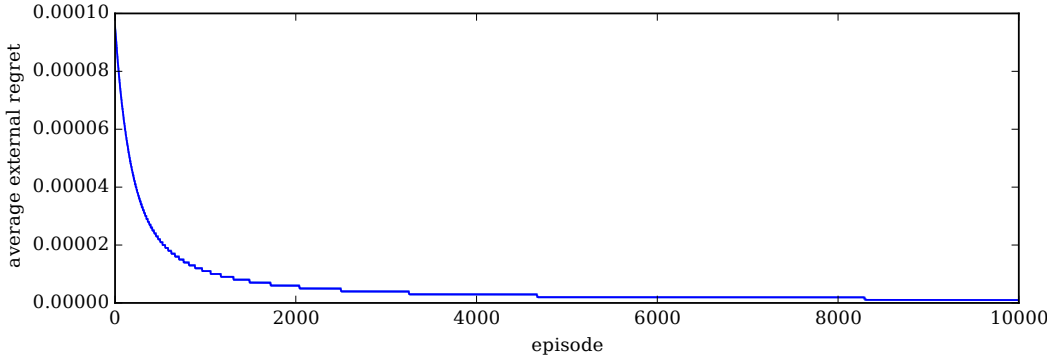
Figure 3.3: External regret along episodes in selected networks, with shaded lines representing the standard deviation.



(a) BB^7 network



(b) OW network



(c) SF network

decreases along time. However, as regret is averaged over time, it takes time for old reward observations to be diluted in the regret.

Last but not least, we can compare the regret bound of our algorithm to that achieved by our approach. Such an upper bound can be computed by applying the parameters used in the experiments to the bound presented in Theorem 3.6. For instance, in the case of the B^1 network, we have[7] that $K = |A| = 3$, $\mu = 0.99$, and $T = 10,000$,

[7]Observe that, in Theorem 3.6, the symbol K stands for $|A|$ (i.e., the number of *all* possible actions) not for the parameter used to run the KSP algorithm. In the road networks considered here, this number can be arbitrarily high and. In fact, computing the set of all possible routes is a complex problem per se. In this sense, here we compute the regret bound for each network using parameter K (i.e., one the used to limit the number of routes). We remark that this is not a restrictive assumption however: the lower the value of K,

which results in an upper bound of 0.0066. Similarly, we can obtain the following upper bounds for the other Braess networks: 0.0149 (for the B^2 network), 0.0074 (for B^3), 0.0074 (for B^4), 0.0174 (for B^5), 0.0174 (for B^6), 0.0174 (for B^7), 0.0024 (for BB^1), 0.0149 (for BB^3), 0.0149 (for BB^5), 0.0149 (for BB^7), 0.0174 (for OW), 0.0374 (for SF). We remark that the regret bound is more sensitive to the μ decay rate than to the other parameters. As a consequence, the regret bound is lower for the networks that fared better with lower decays. Hence, as expected, the experimental results show that the regret achieved by our method is consistent with the bound defined in Theorem 3.6.

Therefore, the experiments confirm our theoretical results, showing that our approach is no-regret and that it approaches the UE. Consequently, we can confirm our initial hypothesis, stating that the use of regret as reinforcement signal leads reinforcement learning agents to converge to the user equilibrium.

3.5 Related work

According Blum and Mansour (2007), regret is useful for analysing the performance of agents that repeatedly need to make decisions. In general terms, this applies to game theory, learning theory, online optimisation, and so on. In the particular context of reinforcement learning, regret has been typically used as a measure of convergence (SHOHAM; POWERS; GRENAGER, 2007; BUŞONIU; BABUSKA; SCHUTTER, 2008). In contrast, in this chapter we used regret to explicitly *guide* the learning process. So far, only a few works investigated such direction. Hart and Mas-Colell (2000) proposed one of the first approaches to employ regret as the learning signal, but using an alternative regret formulation aiming at the correlated equilibrium. Bowling (2005) devised the no-regret GIGA-WoLF algorithm, but it only applies to 2-player-2-action games. Banerjee and Peng (2005) proposed a no-regret algorithm with fewer assumptions on the problem structure, but regret was not employed to guide the learning process. Zinkevich et al. (2008) and Waugh et al. (2015) minimised regret in extensive form games. However, they assume that the regret of all possible actions is known by the agents in advance. Recently, Prabuchandran, Bodas and Tulabandhula (2016) aimed at minimising the cumulative regret, but they assumed that the optimal policy structure is known. In this chapter, we considered another direction, employing regret to guide the learning process. We remark that, by definition, computing regret exactly requires the reward of all actions

the lower the regret bound. Thus, the bounds computed using K are upper bounded by those using $|A|$.

along time, which is not available to the agents. Thus, we have shown how such values can be estimated by the agents.

Regret was also employed in discrete choice models to improve predictions of travellers' behaviour. Chorus and colleagues (2008, 2010) proposed the random regret minimisation (RRM) model, which asserts that travellers try to avoid regret when making decisions. Chorus (2012) revisited regret theory assumptions by considering that choice behaviour is affected both by regret and utility. Later on, Ben-Elia, Ishaq and Shiftan (2013) investigated how travellers' experiences affect their regret and behaviour. The regret theory model was also investigated in terms of the stochastic UE by Li and Huang (2017). Importantly, though, these models (unlike ours) do not take adaptation into account and assume that drivers have full knowledge regarding their regrets and/or travel costs distributions. Moreover, the travel costs are frequently assumed to be fixed. Notwithstanding, we remark that the costs change steadily as a consequence of the agents' learning/adaptation process, which makes the agents' objective a moving target. For these reasons, such models are not suitable for investigating how *individual* agents interact and learn to maximise their rewards.

Congestion games (ROSENTHAL, 1973; ROUGHGARDEN, 2005) is another framework to address route choice. Chien and Sinclair (2007) and Fischer, Räcke and Vöcking (2010) proposed methods for accelerating the equilibrium computation. However, they assumed that only a single agent can change its route per time step. Chan and Jiang (2016) proposed a compact, tree-based representation of the problem. However, their method's efficiency strongly depends on the network topology. Blum, Even-Dar and Ligett (2010) guaranteed the convergence of routing games to an approximate UE when all agents are using no-regret strategies. However, in contrast to our work, they neither investigate *how* agents can obtain such no-regret strategies nor if such strategies indeed exist. In contrast, in this chapter, we *proposed a regret-minimising RL approach* and formally *proved its convergence to an approximate UE*, without relying on previous works' assumptions. Furthermore, we represent flows as integers (which correspond to agents), whereas congestion games literature assume infinitesimal flows (which simplify the convergence analysis).

The regret of route choice has also been approached in the online optimisation literature. The agent's feedback can be transparent (ZINKEVICH, 2003) or opaque (AUER et al., 2002; AWERBUCH; KLEINBERG, 2004), where the environment reveals the reward of all routes or of the taken route, respectively. The latter is equivalent to the

route choice problem and was first investigated by Awerbuch and Kleinberg (2004), who bounded the regret to $O(T^{2/3})$. However, they assumed the reward functions are constant and defined a priori, regardless of the current environment state. Dani, Kakade and Hayes (2007) later improved such a bound to $O(\sqrt{T})$, but lacking an efficient algorithm. Abernethy, Hazan and Rakhlin (2012) achieved a regret of $O(\sqrt{T \log T})$ with an efficient algorithm, but assuming the reward functions are constant and defined a priori. Agarwal, Dekel and Xiao (2010) improved the bound to $O(\sqrt{T})$ in expectation, but for the multi-point version of the problem, in which the agent can observe its reward at any point. Zhang et al. (2015) considered more general reward functions, but assuming they are monotonically increasing (given the flow of vehicles). Here, we achieved a bound of $O\left(\left(\frac{K-1}{TK}\right)\left(\frac{\mu^{T+1}-\mu}{\mu-1}\right)\right)$ without relying on the assumptions of previous works. Moreover, we provided a simple, yet efficient algorithmic solution that provably approximates the UE.

Recent works proposed alternative regret formulations. Arora, Dekel and Tewari (2012) presented the *policy regret*, which considers the effect of actions as if they were taken. However, no one could potentially obtain such information in traffic domains. Heidari, Kearns and Roth (2016) employed the concept of policy regret to address the multi-armed bandit problem. Zinkevich et al. (2008) introduced *counterfactual regret* to estimate the regret in extensive form games with imperfect information. In this chapter, we presented the *action regret*, which measures the regret of individual actions. Furthermore, in contrast to previous approach, we have shown how such action regret can be estimated by the agents using only local information. A similar formulation was presented by Baird (1994). However, their formulation can only be computed by the agents if partial knowledge on the reward functions is available.

3.6 Discussion

We introduced a regret-minimising reinforcement learning approach through which agents can learn how to choose their best routes. In route choice, agents can only observe the reward of the taken actions. In our approach, agents estimate the reward of their actions based on previous observations and use such estimates to compute the (action) regret associated with their actions. Using this formulation, we embodied each agent with the Q-learning algorithm with decaying learning and exploration rates. The idea underlying our approach is that agents can estimate the regret of their actions so as to choose the ones

which present lower regret, which translates into choosing the ones with highest rewards.

We provided theoretical and experimental results. In the theoretical side, we proven that our approach minimises the agents' regret in the limit. Specifically, we bounded the agents' regret to $O\left(\left(\frac{K-1}{TK}\right)\left(\frac{\mu^{T+1}-\mu}{\mu-1}\right)\right)$ after T timesteps, where K is the number of available routes and μ is the decay rate of the exploration parameter. Moreover, we proved that the system converges to an approximate user equilibrium. On the experimental side, we validated our theoretical results in several road networks available in the literature, achieving an average proximity to the UE of 99.38% and an average regret of only 0.0049. Together, our results confirm our initial hypothesis, showing that using the actions' estimated regret as reinforcement signal leads reinforcement learning agents to converge to the UE.

4 THE ROLE OF TRAVEL INFORMATION

In the previous chapter, we considered how agents can estimate their regret and use it to guide their learning process. As a natural extension, this chapter presents an investigation on how the agents' performance is affected by the provision of travel information. Specifically, we introduce a (mobile) navigation entity that provides travel information to the agents, and extend the approach of previous chapter so that agents can compute better estimates of their regret. Through experiments, we show that the travel information improves the agents' performance, should it be available.

4.1 Motivation and contributions

In general, as seen in Chapter 2, regret cannot be computed (and used) by agents because its calculation requires observing the costs of all available routes (including the non-taken ones). In order to use regret, most existing approaches assume that it is known a priori by the agents. In contrast, in Chapter 3 we presented a method for the agents to estimate such regret using only local knowledge. In this chapter, we go beyond and consider the case where *travel information* is also available to the agents. The use of this kind of information has become possible through the use of on-line services (e.g., Waze, Google Maps), which provide travel information to end-users through mobile navigation apps (VASSERMAN; FELDMAN; HASSIDIM, 2015; HASAN et al., 2016). Nonetheless, instead of assuming that such services have full knowledge and provide the real regret to the agents, we simply assume that these services have a possibly more precise information on the routes' travel times than that of agents.

In this chapter, we extend the approach presented in Chapter 3 to take non-local information into account. Specifically, we propose a method for agents to estimate their regret using both local information (an internal history of observed rewards) and global information (travel times provided by a mobile navigation entity, *app* henceforth). In this sense, when estimating an action's regret, the agent considers not only the action's estimated travel time, but also the travel time reported by the app. The rationale behind using app information is that agents can better estimate the regret of their actions. As such, our hypothesis is that the use of non-local information improves agents' performance.

In particular, the contributions of this chapter are:

- We define a (mobile) navigation entity (the app) that provides travel information[1] to the agents. Information here is simply the average travel times of the routes used by the agents. Such information is useful for the agents to estimate their regret.
- We introduce a method for agents to *estimate* their action regret using a linear combination of their experience (rewards received in previous episodes) and information provided by the app. We show that such estimates can be used to improve the learning process.

4.2 Learning with improved estimates of action regret

This section presents our method for the agents to learn to choose their best routes by minimising regret. Initially, we introduce the app and its travel recommendations (Section 4.2.1). Then we discuss how the agents can estimate the regret of their actions using local and non-local information (Section 4.2.2) and present an algorithmic solution for them to learn using such estimates (Section 4.2.3).

In our approach, apart of the drivers (which are represented as Q-learning agents), we also model an entity (the *app*) responsible for providing travel information to the agents. The travel information here is simply the average travel times (based on past episodes) of an agent's *routes*. We remark that the app is not necessarily an agent (indeed, this is not relevant to the purposes of this work).

Our approach works as follows. At the beginning of a learning episode, the app estimates the average travel time of every route based on previous episodes. This information is provided to the agents (each driver receives the estimated travel time of its routes). Afterwards, each agent takes an action and estimates its regret using the received reward *and* the app information for that action. As an intermediate step, each agent also estimates the costs of its non-taken routes, which is required for computing the regret of other actions. Observe that agents use the app information indirectly, i.e., for computing regret, not for choosing the action. Algorithm 4.1 presents a sketch of the proposed method.

[1] We interchangeably refer to the app's travel information as *recommendations* hereinafter.

Algorithm 4.1: Regret-minimising Q-learning with app information (for agent i)

input: set of actions A_i, learning decay rate λ, exploration decay rate μ, number of episodes T

initialise Q-table: $Q(a_i) \leftarrow 0 \; \forall a_i \in A_i$;
initialise history of estimates: $H_i \leftarrow \emptyset$;
for $t \in \{1, \ldots, T\}$ **do**
  $\alpha \leftarrow \lambda^t$; $\epsilon \leftarrow \mu^t$; `// update learning and exploration rates`
  $\{\hat{r}(a_i^t) \mid a_i \in A_i\} \leftarrow$ receive app recomendations;
  $\dot{a}_i^t \leftarrow \epsilon$-greedy; `// choose (and take) action using ϵ-greedy`
  $f_{\dot{a}_i^t} \leftarrow$ observe travel time on $\dot{a}_i^t$;
  $r(\dot{a}_i^t) \leftarrow -f_{\dot{a}_i^t}$; `// compute the reward of` $\dot{a}_i^t$
  for $a_i^t \in A_i$ **do**
    $\tilde{r}(a_i^t) \leftarrow \begin{cases} r(a_i^t) & \text{if } a_i^t = \dot{a}_i^t \\ \tilde{r}(a_i^{t-1}) & \text{otherwise} \end{cases}$; `// update estimate` $\tilde{r}(a_i^t) \in H_i$
  end
  `// compute regret of` $\dot{a}_i^t$
  $\tilde{\mathcal{R}}^T_{i,\dot{a}_i^t} \leftarrow \max_{b \in A_i} \frac{1}{2}\left[\hat{r}(b_i^t) + \frac{1}{T}\sum_{u=1}^{T} \tilde{r}(b_i^u)\right] - \frac{1}{T}\sum_{u=1}^{T} \tilde{r}((\dot{a}_i^t)_i^u)$;
  $Q(\dot{a}_i^t) \leftarrow (1-\alpha)Q(\dot{a}_i^t) + \alpha\tilde{\mathcal{R}}^t_{i,\dot{a}_i^t}$; `// update Q-value of` $\dot{a}_i^t$
end

4.2.1 The app

The *app* is an entity responsible for providing travel information to the agents. Travel information here refers to the *average travel time associated with each route* of an agent. We assume the app has access to the true travel time[2] of all routes within the network. By employing such information, the app can compute the average travel time of each route, which is then provided to the agents that could potentially use that given route. We remark that although computing such information is not trivial, existing on-line services already provide this kind of information to end-users through mobile apps, such as Waze and Google Maps. In these apps, the user specifies a destination and, based on its origin, the apps suggest the best routes at that given moment.

In our approach, the app suggestions are used in a slightly different way than in existing mobile apps. We focus neither on how the app works nor on how much the users adopt its suggestions (in fact, there are several works on this line, as seen in Sec-

[2]Recall that the travel time on a route is computed using Equation (2.2), which depends on the travel time of the links comprising it. By assuming that the app has access to the true travel time of all routes, it follows that it has access to the links' travel times as well (where non-taken links have free flow travel time). Hence, the app can even infer the travel times of non-taken routes.

tion 2.1.3). Rather, we simply assume that agents do receive route information but take their decisions based on their overall experience, which is encoded in their Q-tables. The recommendation is only used to compute the agents' regret. After all, the agents may regret not following the recommendation. The rationale here is that, although the recommended routes may be appealing, the users may prefer to follow their own knowledge. Nevertheless, the app recommendations are indirectly considered in the agents' decisions process, since the regret is used to update the Q-tables. In other words, we are only interested in how the app information may improve the agents' regret.

Given a route $a \in A$, let $r(a^t)$ be the reward for taking that route at time t, as formulated in Equation (2.5). We assume the app can observe the reward of all actions at the end of each episode[3]. Using such information, the app can compute the average reward of each route. Precisely, the average reward of action a up to time t is computed as

$$\hat{r}(a^t) = \frac{1}{t} \sum_{u=1}^{t} r(a^u).$$

Then, at the beginning of each episode t, the app provides to agent i the average travel time for all of its routes, i.e., the set $\{\hat{r}(a^t) \mid a \in A_i\}$.

4.2.2 Estimating Regret

In this section, we detail how an agent can estimate its regret by combining local information (i.e., the rewards it actually observed) and global information (i.e., the average estimated reward of all routes, as provided by the app). The local information is computed in the same way as described in Section 3.2.1. The global information, on the other hand, is simply obtained through the app, as detailed in Section 4.2.1.

The *estimated action regret* of action a for agent i up to time T can then be reformulated as in Equation (4.1). The estimated action regret defined here is similar to the one presented in Chapter 3. Intuitively, Equation (4.1) can also be seen as an estimate of the average amount lost by agent i up to time T for taking action a (latter term) rather than the best estimated action (former term). The difference is that the first term of the

[3]Recall that the route choice problem is modelled as a commuting scenario where daily (i.e., episodic) trips are made under approximately the same conditions (i.e., same demand and OD pairs). Hence, only episodes are relevant here, not time of the day. On the other hand, the travel time on links changes from one day to another since drivers may change their route choices. This explains why agents need to learn which route is the best.

equation now comprises a linear combination of the local average cost (using $\tilde{r}(a)$) and global average cost (using $\hat{r}(a)$ from the app). In this sense, the agent can now obtain more precise estimates of the reward associated with non-taken actions.

$$\tilde{\mathcal{R}}_{i,a}^{T} = \max_{b_i^t \in A_i} \frac{1}{2} \left[\hat{r}(b_i^t) + \frac{1}{T} \sum_{t=1}^{T} \tilde{r}(b_i^t) \right] - \frac{1}{T} \sum_{t=1}^{T} \tilde{r}(a_i^t) \tag{4.1}$$

In the same way, we can reformulate Equation (2.8) to obtain the *estimated external regret* of agent i according to Equation (4.2). The estimated external regret $\tilde{\mathcal{R}}_i^T$ of agent i expresses how much worse it performed, on average, up to time T for not taking only the best action regarding its experience.

$$\tilde{\mathcal{R}}_{i}^{T} = \max_{a_i^t \in A_i} \frac{1}{2} \left[\hat{r}(a_i^t) + \frac{1}{T} \sum_{t=1}^{T} \tilde{r}(a_i^t) \right] - \frac{1}{T} \sum_{t=1}^{T} r(\dot{a}_i^t) \tag{4.2}$$

As discussed in Section 3.2.1, we remark that the above regret formulation could be extended in several ways. For instance, a weighted (rather than linear) combination of the local and global average costs could benefit the agents in the first episodes, when they have less experience. In fact, investigating how such weight correlates with a possible improvement in the agents' performance represents an interesting research direction. In this thesis, however, we leave other formulations as future work and concentrate on the linear combination case, which is simpler and paves the way to more elaborate formulations.

4.2.3 Learning to Minimise Regret

Building upon the regret estimations from the previous section, we now present an RL algorithm enabling the agents to learn a no-regret policy. The sketch of our method is presented in Algorithm 4.1. Every driver $i \in D$ is represented by a Q-learning agent. The route choice problem can then be modelled as a stateless MDP. Let $A_i = \{a_1, \ldots, a_K\}$ be the set of routes of agent i. The set of agents' actions is denoted by $A = \{A_i \mid i \in D\}$. Observe that if two agents i and j have the same OD pair, then $A_i = A_j$. The reward for taking action a at time t is $r(a^t)$, as given by Equation (2.5). We remark that the reward function $r(a^t)$ is not deterministic. Instead, it varies on time depending on the current state of the road network (i.e., which routes other agents are taking).

The learning process works as follows. At each episode $t \in [1, T]$, each agent

$i \in D$ receives[4] recommendations $\{\hat{r}(a_i^t) \mid a_i \in A_i\}$ from the app (line 5 of Algorithm 4.1). Agent i then chooses an action $\dot{a}_i^t \in A_i$ using the ϵ-greedy strategy (based on the actions' Q-values). After taking the chosen action, the agent receives a reward of $r(\dot{a}_i^t)$. Afterwards, the agent updates its history H_i using Equation (3.1) (lines 9–11 of Algorithm 4.1) and calculates the estimated regret of action $\dot{a}_i^t$ using Equation (4.1) (line 12 of Algorithm 4.1). The recommendations received in the beginning of the episode are used at this point. Finally, the agent updates $Q(\dot{a}_i^t)$ *using the estimated action regret for that action* (line 13 of Algorithm 4.1), as defined in Equation (4.3). The learning (α) and exploration (ϵ) rates are initialised with value 1.0 and are multiplied by the decay rates (λ and μ) at each episode so that their value at a given episode t is $\alpha(t) = \lambda^t$ and $\epsilon(t) = \mu^t$, respectively. This process is repeated for each episode.

$$Q(\dot{a}_i^t) = (1 - \alpha)Q(\dot{a}_i^t) + \alpha\tilde{\mathcal{R}}^t_{i,\dot{a}_i^t} \tag{4.3}$$

Recall that the estimated action regret guides the learning process (i.e., for updating an agent's policy), which, as discussed in Section 3.2.2, leads an agent to minimising its estimated external regret. Moreover, we emphasise that this is only possible because the action regret formulation of Equation (4.3) decomposes the regret per action, thus allowing an agent to evaluate how much a particular action contributes to its regret.

As in the previous chapter, we highlight that Algorithm 4.1 is an abstract scheme of the algorithm from the agent's perspective. We refer the reader to Appendix A for the complete details on the simulation procedure. The time and space complexity of our approach are presented in the next proposition. The proof is also detailed in the appendix.

Proposition 4.1. *Our regret-minimising Q-learning with app information approach has* $O(T(dK + ld + lmK))$ *time complexity and* $O(dK)$ *space complexity, for* T *episodes,* d *drivers, and* K *actions,* l *links, and* m *OD pairs.*

4.3 Experimental evaluation

This section provides empirical results and analysis regarding our method's performance. The main hypotheses to be validated here are that: (i) the use of action regret (with app information) as reinforcement signal leads RL agents to converge to an approx-

[4]Note that the received recommendations are not necessarily used. Instead, as discussed in Section 4.2.2, such recommendations are used by the agent to improve its estimates of the actions' regret.

Table 4.1: Characteristics of the networks used for validation of our approach.

Network	Nodes	Links	OD pairs	Number of drivers	*avg-tt*[a] under UE
B^1	4	5	1	4,200	20.00
B^2	6	9	1	4,200	30.00
B^3	8	13	1	4,200	40.00
B^4	10	17	1	4,200	50.00
B^5	12	21	1	4,200	60.00
B^6	14	25	1	4,200	70.00
B^7	16	29	1	4,200	80.00
BB^1	8	8	2	4,200	10.00
BB^3	12	16	2	4,200	22.00
BB^5	16	24	2	4,200	50.30
BB^7	20	32	2	4,200	≈123.84
OW	13	48	4	1,700	≈67.16
SF	24	76	528	360,600	20.76

[a] Values reported in the literature (STEFANELLO; SILVA; BAZZAN, 2016; STEFANELLO; BAZZAN, 2016).

imate UE, and (ii) the agents' regret is reduced when the app-based information is used. Recall that *learning* means finding the best route to take and that *convergence* refers to a point at which the agents keep *exploiting* their knowledge most of the time and the system is somewhat *stable* (i.e., agents only observe small fluctuations in their costs). Our key contributions is to show that, using our approach, such a stable point is close to the UE.

4.3.1 Methodology

We simulate our approach as described in Appendix A. In the simulations, we use the same road networks[5] used in Chapter 3. The reader is referred to Section 3.4.1 for the complete description of these networks. In order presentation, nevertheless, here we replicate the table summarising these networks in Table 4.1. The most representative such networks are illustrated in Appendix B.

The methodology employed here is the same used in the previous chapter. In order to avoid arbitrarily large sets of actions, we limit the routes to the K shortest ones[6] using the KSP algorithm (YEN, 1971). As before, the best value for K (i.e., the one which produces the best results) depends on the networks characteristics, thus varying from one

[5]The road networks are available at <https://github.com/maslab-ufrgs/network-files>.

[6]As for the BB networks, among the shortest routes we included the one with the least number of links, otherwise the UE would not be possible (STEFANELLO; BAZZAN, 2016). This is a limitation of BB networks. However, we note that such a decision is realistic given that the shortest route is usually considered by drivers even if it is not the fastest one.

instance to another.

An experiment corresponds to a complete execution, with T =1,000 episodes (except for the SF network, in which case we set T =10,000 episodes) episodes of our method on a single network. When an execution is completed, we measure its performance by means of the average travel time (*avg-tt* hereinafter, measured in minutes), the average external regret, and the average proximity to the UE (computed using Equation (3.11)). The results reported for each of the above measures are given considering the last episode.

We tested different value combinations for the Q-learning parameters (the tuning process is discussed in Section 4.3.2). For each such combination, 30 repetitions were performed. Our results are compared against standard Q-learning (*stdQL*, hereinafter), which uses reward (rather than action regret) as reinforcement signal. We also include our algorithm without the app (i.e., the one presented in Chapter 3) in the comparisons. In what follows, any claim about whether one approach is better than the other is supported by Student's t-tests at the 5% significance level, except if otherwise stated.

The algorithms, data analysis and plots were all implemented in Python 2.7.

4.3.2 Parameter tuning

The parameters of this approach were tuned here in the same way as in previous chapter. We test different number of routes, namely, $K \in \{4, 8, 12, 16\}$ (intermediate values for K pose no significant difference on the results). The learning and exploration rates, α and ϵ, are initialised with 1.0 and multiplied by their decay rates after each episode. For these decay rates, we tested values $\{0.98, 0.99, 0.995, 0.999\}$, with $\lambda = \mu$. In the case of the SF network, we tested $\lambda \in \{0.9995, 0.9997, 0.9999\}$ and $\mu \in \{0.995, 0.997, 0.999\}$ to account for its higher demand (i.e., its number of vehicles is two orders of magnitude higher than in the other networks). Again, λ was tested with higher values than μ to ensure that agents keep learning even after exploration is decreased.

Table 4.2 presents the best-performing combinations of parameters. As expected, the best value for the parameters is not affected by the app information. The reason is that this information is only used to improve agents' estimates on their routes. The optimisation process itself, on the other hand, remains the same. In general, therefore, the larger the network, the higher the values required for the parameters. The results of the above configurations were selected for further analyses in the next subsection.

Table 4.2: Parameters' configuration that produced the best results for each network.

Network	K	λ	μ
B^1	4	0.99	0.99
B^2	4	0.995	0.995
B^3	4	0.99	0.99
B^4	4	0.99	0.99
B^5	8	0.995	0.995
B^6	8	0.995	0.995
B^7	8	0.995	0.995
BB^1	4	0.98	0.98
BB^3	4	0.99	0.99
BB^5	4	0.995	0.995
BB^7	4	0.995	0.995
OW	8	0.995	0.995
SF	4	0.9999	0.998

4.3.3 Results

The performance of our approach in terms of proximity to the UE and external regret in all evaluated road networks is presented in Tables 4.3 and 4.4, respectively. Results represent averages over 30 repetitions, with standard deviations shown in parentheses. The values of the algorithms' parameters are those listed in Table 4.2. In order to better evaluate our approach, we also show results for our approach using *no* app information (ours–app) and for standard Q-learning (sdtQL).

As seen in the tables, our approach outperforms standard Q-learning on average. In terms of (absolute) proximity to UE, our approach improves[7] upon standard Q-learning by 0.55% (without the app) and 0.57% (using the app) on average. Such results account for an average *decrease in the relative distance to the UE* of 48% (without the app) and 50% (using the app) as compared to standard Q-learning. As for regret, our approach decreases the average regret by 21.5% (without the app) and by 33.9% (using the app). We highlight that, although improving regret does not necessarily translates into higher system's performance, it leads to stronger results, given that it better fits the no-regret property. In other words, improving regret means increasing convergence probability. Therefore, our results confirm one of our initial hypotheses, namely that learning to minimise regret

[7]We highlight that even a 1% improvement in the absolute proximity to the UE represents a significant result here. In fact, state-of-the-art traffic approaches frequently only achieve small improvements as compared to previous works. A representative example here is that of Bar-Gera (2010)'s TAPAS algorithm (which approaches a setting similar to ours, namely the traffic assignment problem), whose absolute improvement in the UE was smaller (10^{-6}%, as compared to previous works) than that achieved by our approach (1.1%, as compared to standard Q-learning).

Table 4.3: Average proximity to the UE (with standard deviation in parentheses) of our approach, with (Ours + app) and without app (Ours – app), and of standard Q-learning (StdQL) on different networks.

	Proximity to the UE		
Network	**Ours+app**	**Ours–app**	**StdQL**
B^1	0.9999 (10^{-5})	1.0000 (0.000)	0.9250 (10^{-2})
B^2	0.9986 (10^{-4})	0.9981 (10^{-4})	0.9570 (10^{-2})
B^3	0.9999 (10^{-5})	0.9999 (10^{-5})	0.9974 (10^{-3})
B^4	0.9999 (10^{-5})	0.9999 (10^{-6})	0.9999 (10^{-6})
B^5	0.9942 (10^{-4})	0.9916 (10^{-4})	0.9889 (10^{-3})
B^6	0.9999 (10^{-4})	0.9994 (10^{-4})	0.9996 (10^{-4})
B^7	0.9998 (10^{-4})	0.9999 (10^{-4})	0.9999 (10^{-5})
BB^1	1.0000 (0.000)	1.0000 (0.000)	1.0000 (0.000)
BB^3	0.9988 (10^{-4})	0.9991 (10^{-4})	0.9941 (10^{-3})
BB^5	0.9985 (10^{-4})	0.9991 (10^{-4})	0.9959 (10^{-3})
BB^7	0.9984 (10^{-4})	0.9979 (10^{-4})	0.9986 (10^{-4})
OW	0.9997 (10^{-4})	0.9997 (10^{-4})	0.9989 (10^{-4})
SF	0.9343 (10^{-4})	0.9344 (10^{-4})	0.9886 (10^{-4})
Average	0.9940 (10^{-2})	0.9938 (10^{-2})	0.9880 (10^{-2})

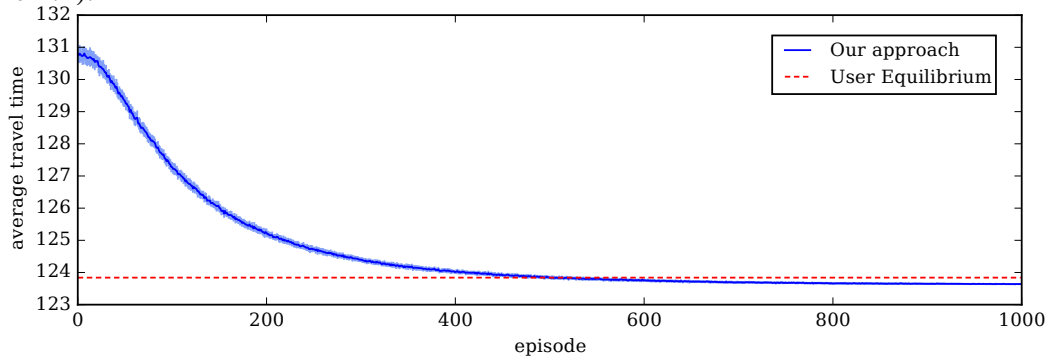
Table 4.4: Average external regret (with standard deviation in parentheses) of our approach, with (Ours + app) and without app (Ours – app), and of standard Q-learning (StdQL) on different networks.

	External regret		
Network	**Ours+app**	**Ours–app**	**StdQL**
B^1	0.0057 (10^{-5})	0.0057 (10^{-5})	0.0121 (10^{-3})
B^2	0.0033 (10^{-5})	0.0034 (10^{-5})	0.0111 (10^{-3})
B^3	0.0021 (10^{-5})	0.0020 (10^{-4})	0.0041 (10^{-3})
B^4	0.0002 (10^{-5})	0.0004 (10^{-4})	0.0010 (10^{-4})
B^5	0.0019 (10^{-5})	0.0025 (10^{-4})	0.0039 (10^{-4})
B^6	0.0019 (10^{-5})	0.0020 (10^{-5})	0.0030 (10^{-5})
B^7	0.0011 (10^{-5})	0.0025 (10^{-4})	0.0019 (10^{-5})
BB^1	0.0016 (10^{-5})	0.0016 (10^{-5})	0.0016 (10^{-5})
BB^3	0.0089 (10^{-5})	0.0173 (10^{-5})	0.0178 (10^{-4})
BB^5	0.0066 (10^{-5})	0.0066 (10^{-5})	0.0063 (10^{-5})
BB^7	0.0034 (10^{-5})	0.0035 (10^{-5})	0.0029 (10^{-5})
OW	0.0152 (10^{-4})	0.0161 (10^{-4})	0.0131 (10^{-5})
SF	1×10^{-6} (0.0)	1×10^{-6} (0.0)	2×10^{-6} (0.0)
Average	0.0040 (10^{-3})	0.0049 (10^{-3})	0.0061 (10^{-3})

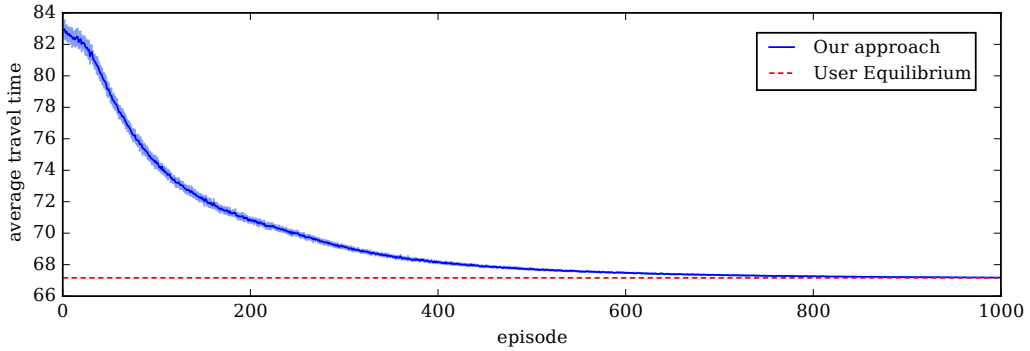
leads the agents to an approximate UE, even when the app information are used.

We can analyse the agents' learning behaviour in more detail by considering the variation of average travel time and external regre along time, as shown in Figures 4.1 and 4.2, respectively. In the plots, each curve is an average over 30 repetitions (with standard

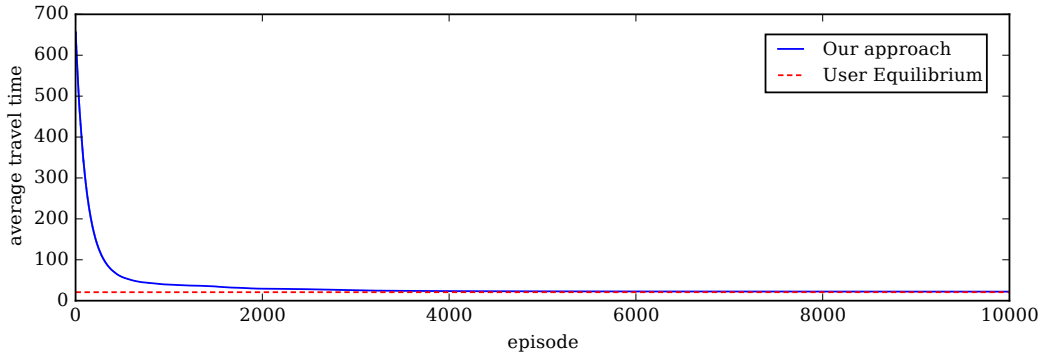
Figure 4.1: Average travel time along episodes in selected networks, with shaded lines representing the standard deviation and dashed lines representing the UE (as reported in Table 4.1).



(a) BB^7 network



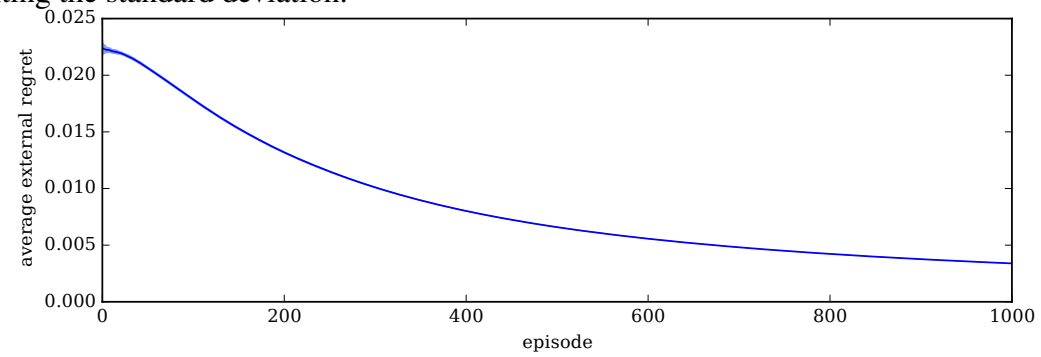
(b) OW network



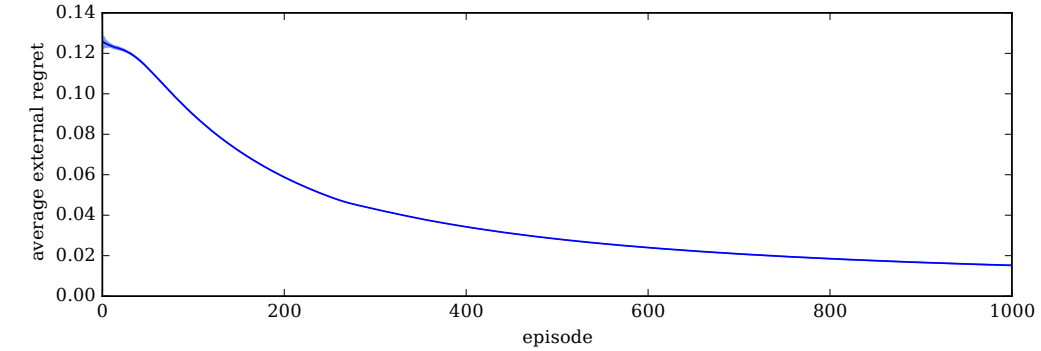
(c) SF network

deviation shown as shaded lines) and dashed lines present the average travel times under UE (as reported in Table 4.1). We highlight that the apparent faster convergence in the SF network (Figures 4.1c and 4.2c) is actually due to the higher number of episodes used for that network, thus concentrating the initial steps in the left side of the plot. As seen in the plots, *avg-tt* is high in the beginning (because of the Q-values initialisation) and decreases steadily as agents explore and learn their routes. In general, the use of the app information pose no significant difference in the agents' learning behaviour itself (as compared the plots of the previous chapter, where no app information was used). In fact, that similarity meets the expectations, given that the problem itself does not change: the

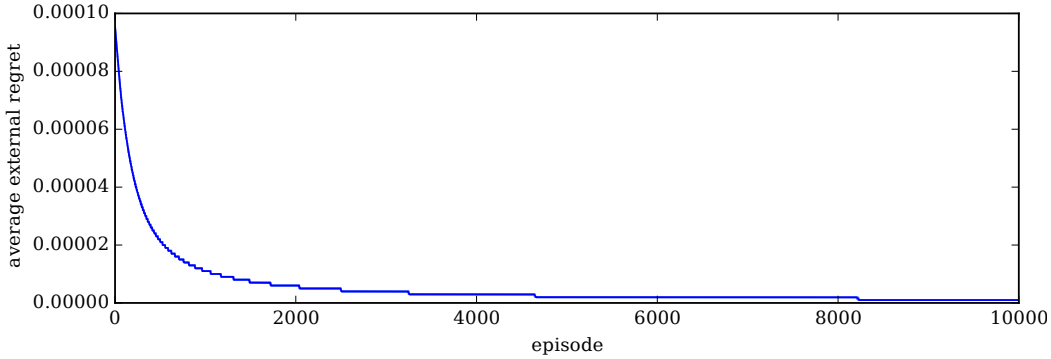
Figure 4.2: External regret along episodes in selected networks, with shaded lines representing the standard deviation.



(a) BB^7 network



(b) OW network



(c) SF network

only difference is that, by using the app, agents now obtain reasonable estimates on their routes' costs sooner.

Our remaining hypothesis states that agents benefit by considering the app's recommendations when computing their regret estimates. The results of this analysis are also presented in Tables 4.3 and 4.4. As seen, the app improves drivers' performance in most cases. When the app is being used, the (absolute) *proximity* to UE is improved, on average, by only 0.02% as compared to the cases where the app is *not* used. This accounts for an average decrease in the relative *distance* to the UE of 3.8%. Moreover, due to the app information, the agents' regret is reduced by 13.7% as compared to when agents are

not using the app. The rationale here is simple: the information provided by the app is more accurate than that estimated by the agents, especially in the initial episodes. Observe that the app's recommendations are only implicitly taken into account when agents make decisions. The point is that agents choose actions based on their Q-values. The app's recommendations, however, are used to *improve regret estimates*, which, in turn, are used to update the agents' Q-tables. In this sense, recommendations play an important role while agents are learning and exploring. On the other hand, when agents' estimates are accurate, app recommendations may be irrelevant. We can then state that using app recommendations lead to reasonable improvements in the agent's performance.

Therefore, we conclude that agents are able to estimate their regret *locally* and to use such information to learn their best routes. But mainly, we could observe that these improvements are even higher when more accurate information is provided (in this case, by means of the app) to the agents. The presented results thus validate our initial hypotheses, namely that (i) the use of action regret (with app information) as reinforcement signal leads RL agents to converge to an approximate UE, and that (ii) agents' regret is reduced when the app-based information is used.

4.4 Related work

The literature on providing and employing non-local information into the agents' decision process is very broad. Here we discuss two kinds of approaches, where the information an agent receives is obtained: from a central authority, and by communicating with other agents. The interested reader is referred to Zhang et al. (2011) and Essen et al. (2016) for a more detailed review.

In a setting similar to ours, Vasserman, Feldman and Hassidim (2015) proposed the use of a Waze-like app that recommends different routes to drivers, aiming at efficiently spreading traffic over a road network. They assumed that the app can observe the agents' routes *before* trips are actually taken. This allows the app to anticipate a recommendation signal. Notwithstanding, assuming that drivers share their decisions in advance may not be realistic. Furthermore, their analyses are limited to parallel-links networks.

The adoption of advanced traveller information systems (ATIS) has also drawn attention (HALL, 1996). Dell'Orco and Marinelli (2017) considered that the agents' compliance level is proportional to the uncertainty associated with their knowledge. Gao, Frejinger and Ben-Akiva (2010) employed prospect theory to investigate how agents adapt

their routes (i.e., en route decision) when facing travel information. Rashidi et al. (2017) reviewed works that employ social media information to improve transportation planning and management. The impact of different levels of information in the route choice process has also been investigated. The idea here is to understand how drivers react to traffic information. Dia and Panwai (2007), Dia and Panwai (2014) employed neural networks to predict the compliance level of drivers who receive traffic suggestions. Peeta and Yu (2005) developed a fuzzy-based system to predict the behaviour of drivers who are provided with traffic information. Nonetheless, it should be noted that these approaches, as opposed to ours, focus mainly on analysing the impact of the traffic information itself, and neither technique is assessed in terms of the UE.

Klügl and Bazzan (2004) investigated how traffic forecasts impact drivers' decision making in a two-route scenario. They assumed that a traffic control system observes drivers' decisions and use such information to compute a traffic forecast. Afterwards, drivers can change their route choices before actually driving. However, they assume that such control system can observe drivers' actions. Later on, Bazzan and Klügl (2005) investigated the impact of providing biased recommendations to drivers. Specifically, a centralised service suggests routes aiming at a system efficient equilibrium, which may deteriorate the performance of some drivers in favour of others. However, their approach was only tested in the networks that undergo the (BRAESS, 1968)'s paradox.

Ben-Elia, Ishaq and Shiftan (2013) developed a regret-based discrete choice model. In their work, regret is defined in terms of travellers perceptions (of previous trips) and route travel times. However, they assumed that travel times do not change with time. Chorus, Walker and Ben-Akiva (2013) investigated how agents behave when deciding to acquire (or not) travel information and how such decisions (and information received) affect their behaviour. They proposed a discrete choice model in which both aspects (when to acquire information and how to use it) are considered by the agent. Wang, Ma and Jia (2013) proposed a choice model that employs prospect theory and replicator dynamics to describe how drivers' decisions evolve under risk. The impact of inaccurate travel information on choice behaviour was studied by Ben-Elia et al. (2013). Nonetheless, in contrast to our work, all these methods focus on developing discrete choice models. As discussed in Section 2.1.2, recall that such models aim at predicting travellers behaviour to assist traffic managers (i.e., from a centralised perspective) on analysing traffic patterns.

Hasan et al. (2016) proposed a social network through which drivers can exchange traffic information. Drivers participating in the social network (i) report congestion lev-

els on their routes and (ii) use the report from their peers to compute the routes' utility. Notwithstanding, as opposed to our work, agents in their approach do not learn, but only choose the route with the highest utility most of the time. Social networks were also the object of study of Pathania and Karlapalem (2015) and He et al. (2013). In the former, social networks are used to improve demand predictions for metro lines. A transport manager agent then optimises the train schedule so as to meet the predicted demand. However, their work was only applied to public transportation scheduling and did not take drivers' behaviour into account. Following this line, He et al. (2013) proposed a framework for extracting traffic indicators based on social media information. Nonetheless, their framework was not applied to improve traffic conditions.

4.5 Discussion

In this chapter, we investigated how to improve the agent's learning process by providing travel information to them. In particular, we developed a navigation entity (the *app*) that tells the agents the average travel time on their routes. We then extended the work of Chapter 3 using the received information to enhance the agents' estimates over their actions' regret. The idea behind such a formulation is that, when regret estimates are more accurate, the agents tend to better choose their actions. We highlight, however, that the assumptions underlying our app are weaker than those of previous works. Although the app can observe the cost on all routes, the information provided to the agents is simply the average travel time on the routes. Consequently, this information may not be fully accurate. Nevertheless, it is still useful in the agents' learning process.

Our approach was experimentally evaluated in several road networks available in the literature. Based on the experiments, we observed that, when agents use the app recommendation, the average distance to the user equilibrium decreases by 3.8%. Moreover, under these circumstances, the regret is reduced by 13.7%. These results are a consequence of the improvement in the agents' estimates over their regrets. Although agents' decisions are not explicitly related to the app's recommendations, these are used to estimate regret and, thus, to update their Q-tables. Recall that the app's recommendations may not be fully accurate. Nonetheless, such information is usually more accurate than the agent's one, especially when they have not yet experienced enough their actions. Hence, the app's recommendations are particularly important in the beginning of the learning process, where agents knowledge is less accurate.

5 SYSTEM-EFFICIENT EQUILIBRIA

In the previous chapters, we considered how reinforcement learning agents can learn to minimise their regret. Firstly, we have shown that, when agents use regret to guide their learning process, they converge to an approximate user equilibrium (Chapter 3). We then moved forward, analysing the impact of travel information in the learning process, should it be available. However, as seen in Chapter 2, the user equilibrium is inefficient from the system's perspective.

In this regard, in this chapter we investigate under what conditions the system can be guaranteed to convergence to a system-efficient equilibrium. Recall that we use the term *system-efficient equilibrium* to refer to a user equilibrium that is aligned to the system optimum. We advance towards this direction by incorporating some sort of system's performance measure into the agents' utility. This problem can be approached in several ways. Here, we consider a natural extension of regret. Specifically, in this chapter we show how the impact an agent causes on others can be used to penalise its selfish decisions. Incorporating such impact into the agent's regret results in traditional tolling schemes. In this regard, we simplify our approach, *leaving the regret formulation aside* and focusing specifically on the tolling scheme.

5.1 Motivation and contributions

As discussed in Chapter 2, the self-interested behaviour of drivers trying to minimise their travel costs leads to the User Equilibrium (UE). However, although appealing from the drivers' perspective, the UE does not represent the system at its best operation (i.e., when average travel costs are minimum). In fact, the average travel time under UE can be considerably higher than the so-called system optimum (SO). Such a deterioration in the system's performance due to drivers' selfish behaviour is known as the Price of Anarchy (PoA) (PAPADIMITRIOU; TSITSIKLIS, 1987).

The system optimum is only attainable if some agents take sub-optimal routes (from their perspective) in the benefit of the system's performance. In general, however, one cannot assume that agents behave altruistically with respect to the social welfare. Specifically, given that agents act rationally to minimise their own costs, no agent would take a route that improves the system's performance in detriment of its own performance: whenever a better route is available, the agent shall opt for it. Moreover, even if the

agents were altruist, no agent would be able to evaluate how much its decisions impact the system's welfare due to its limited observability about such a performance measure.

Different approaches have been proposed in the literature to overcome the limitations imposed by the agents' selfish behaviour. One of such alternatives refers to assuming that each road has a toll, whose value incentivises agents to take socially optimal routes (BECKMANN; MCGUIRE; WINSTEN, 1956). A particularly relevant tolling scheme here is the marginal-cost tolling (MCT), in which each agent is charged proportionally to the cost (e.g., travel time) it imposes on others (PIGOU, 1920). By employing MCT, the UE is guaranteed to converge to the SO.

In this chapter, we approach the toll-based route choice problem from the multi-agent reinforcement learning (MARL) perspective and provide theoretical guarantees on the agents' convergence to a system-efficient equilibrium (i.e., aligning the UE to the SO). As in Chapters 3 and 4, each driver is represented by a Q-learning agent whose objective is to learn which route minimises its expected cost. We design tolls following the MCT scheme, where the cost of a link comprises two terms: (i) the travel time and (ii) the toll charged on it. We then propose a generalised toll formulation that charges an agent only after it has completed its trip. Our a posteriori tolling scheme allows for the toll values to be computed by the agents themselves. In this sense, as compared to existing approaches—e.g., Sharon et al. (2017)—our formulation is more general (i.e., it applies to most traffic scenarios), it is fairer (i.e., agents pay exactly their marginal costs), and it is easier to deploy (i.e., it has fewer infrastructure requirements). To the best of our knowledge, this is the first time that RL agents are proven to converge to a system-efficient equilibrium without assuming they have full knowledge about the reward functions.

The main contributions of this chapter can be enumerated as follows:

- We generalise the toll values formulation for univariate, homogeneous polynomial cost functions. We show that this formulation comprises the most commonly-used cost functions in the literature.
- We formulate an MCT scheme in which drivers are charged a posteriori, whenever they finish a trip. This formulation allows the tolls to be computed locally by the agents. We show that this scheme is fairer and simpler than a priori schemes. An example on this is presented later, in Section 5.3.2.
- We define an RL algorithmic solution through which each driver computes the toll value it has to pay whenever it finishes its trip, and learns the best route to take.

- We provide theoretical results showing that our method converges to the UE in the limit (as opposed to existing works, which assume that the UE is given) and that, by using MCT, the UE corresponds to the SO. Thus, in the limit, the PoA achieves its best ratio. We also validate these results in different road networks from literature.

5.2 Learning system-efficient equilibria using marginal-cost tolling

This section presents our reinforcement learning method through which agents can compute the tolls associated with their routes and use that information to learn their best routes. As before, we model the problem as a stateless MDP and represent drivers by means of Q-learning agents. At every episode, each such agent chooses a route from its origin to its destination and, once the trip is completed, the agent observes its travel time. Building upon such observations, we propose a general tolling scheme through which the toll values can be computed a posteriori by the agents themselves (Section 5.2.1). Together, the travel time and toll value an agent experiences in a given route compose the cost of such route. Using this cost, each agent then computes the regret associated with the chosen route and uses such information to update its Q-table (Section 5.2.2).

Our generalised tolling scheme assumes that each agent can observe its travel time and compute its toll a posteriori. In practical terms, this is equivalent to coupling each driver with a mobile navigation device, which computes and provides such information (PALMA; LINDSEY, 2011).

We remark that, by definition, travel times and tolls are defined per link, whereas agents' decisions are based on routes. In this sense, hereafter we refer to a route's travel time (and toll value) as the sum of its links' travel times (and toll values).

5.2.1 Generalising toll values

Toll values are defined according to the marginal cost of the agents. Specifically, the toll charged on link l is computed as the product of (i) its flow (i.e., the number of vehicles on it) and (ii) its VDF's derivative, as shown in Equation (5.1).

$$\tau_l = x_l \cdot f_l'(x_l) \tag{5.1}$$

The derivative of the link's cost depends on the VDF being employed. Sharon et al. (2017) have shown that, for the BPR function (Equation (2.3)), the resulting marginal cost toll can be written as

$$\tau_l = \beta(f_l - F_l),$$

with, recall, f_l and F_l representing the *actual* (i.e., as given by the VDF function) and *free flow* (i.e., the lower bound when $x_l = 0$) travel times on link l, and β denoting a constant of the road network instance (as discussed in Section 2.1.1). Their formulation, however, is limited to the BPR function. Notwithstanding, considering that several other VDFs are available in the literature (ORTÚZAR; WILLUMSEN, 2011), we go beyond the work of Sharon et al. (2017) and generalise the toll formulation according to the following proposition.

Proposition 5.1. *The marginal-cost toll value τ_l on any link l with a univariate, homogeneous polynomial VDF function is $\beta(p_1 x_l^{\beta})$, where β and p_1 represent VDF-specific constants.*

Proof. First we analyse the case of linear and polynomial functions. Then, we define the general MCT formulation.

Linear functions are in the form $f_l(x_l) = p_1 x_l + p_0$. We consider two such examples from the literature. The OW function (ORTÚZAR; WILLUMSEN, 2011) is represented as $f_l(x_l) = F_l + 0.02x_l = p_1 x_l + p_0$, with $p_0 = F_l$ and $p_1 = 0.02$ representing VDF-specific constants. The linear Braess functions (STEFANELLO; BAZZAN, 2016) can be represented as $f_l(x_l) = \left(\frac{kci_l}{d}\right) x_l = p_1 x_l + p_0$, with $p_0 = 0$ and $p_1 = \frac{kci_l}{d}$ representing VDF-specific constants.

Polynomial functions can be defined in the form $f_l(x_l) = \sum_{\beta=0}^{n} p_\beta x^\beta$. In this chapter we consider the specific case of univariate (single variable), homogeneous (all terms with the same degree) polynomial functions, which can be written in the simpler form $f_l(x_l) = p_1 x_l^{\beta} + p_0$. Such a subclass of polynomial functions includes VDFs that are well-known in the transportation literature, such as the BPR function (Equation (2.3)). The BPR function is represented as $f_l(x_l) = F_l \left(1 + \alpha \frac{x_l}{C_l}^{\beta}\right) = F_l + x_l^{\beta} \left(\frac{\alpha F_l}{C_l^{\beta}}\right) = p_1 x_l^{\beta} + p_0$, with $p_0 = F_l$ and $p_1 = \frac{\alpha F_l}{C_l^{\beta}}$ representing VDF-specific constants. Note that this polynomial definition generalises over linear and constant functions. Specifically, linear functions correspond to the special case where $\beta = 1$ and constant functions correspond to the special case where $p_1 = 0$.

The MCT of link l is defined as $\tau_l = x_l \cdot (f_l(x_l))'$. By using the definition of

univariate, homogeneous polynomial functions above, we have that $\tau_l = x_l(p_1 x_l^{\beta} + p_0)' = x_l(p_1 \beta x_l^{\beta-1}) = \beta(p_1 x_l^{\beta})$, as required. □

We emphasise that Proposition 5.1 only holds when the VDF is defined as an univariate (i.e., with a single parameter, such as flow), homogeneous (i.e., all terms with the same degree) polynomial. It should be noted, however, that this assumption is not unrealistic, given that the most commonly-used VDF functions in the literature are in this class, such as: BPR, OW, Braess, Pigou, etc. Moreover, the above proposition can be extended to overcome these limitations. Such an extension is left as future work.

From Proposition 5.1, observe that computing toll values requires some parameters, such as the flow of vehicles. Recall that this information may not be directly available to the agents. Fortunately, however, such information can be obtained by means of the agents' travel times. In this regard, we can combine Proposition 5.1 with the formulation of Sharon et al. (2017), thus obtaining the next corollary.

Corollary 5.1. *The toll value on link l can be rewritten as $\tau_l = \beta(p_1 x_l^{\beta}) = \beta(p_1 x_l^{\beta} + p_0 - p_0) = \beta(f_l - F_l)$, considering $F_l = p_0$ and $f_l(x_l) = p_1 x_l^{\beta} + p_0$. In other words, whenever an agent finishes its trip (i.e. a posteriori), it can compute the toll on the corresponding route based on its actual and free flow travel times.*

As seen, agents can compute the tolls associated with their routes knowing neither the reward of all routes nor the actions taken by the other agents. Having defined the toll values, we can rewrite the routes reward function as in Equation (5.2), which follows from Proposition 5.1 and Equations (2.2) and (2.5).

$$\begin{aligned} r(a_i^t) &= -\textstyle\sum_{l \in a_i^t} c_l \\ &= -\textstyle\sum_{l \in a_i^t} f_l + \beta(f_l - F_l) \\ &= -(f_{a_i^t} + \beta(f_{a_i^t} - F_{a_i^t})). \end{aligned} \tag{5.2}$$

5.2.2 Learning process

We can now present our RL algorithm. Again, the problem is represented as a stateless MDP and each driver $i \in D$ as a Q-learning agent. The set of routes of agent i is denoted by $A_i = \{a_1, \ldots, a_K\}$. The reward $r(a_i^t)$ that agent i receives for taking route a_i^t at episode (or timestep) t is given by Equation (5.2). The drivers' objective is to maximise their cumulative reward. An overview of our method is presented in Algorithm 5.1.

Algorithm 5.1: Toll-based Q-learning (for agent i)

input: set of actions A_i, learning decay rate λ, exploration decay rate μ, number of episodes T, free flow F_l for all $l \in L$, and β

1 initialise Q-table: $Q(a_i) \leftarrow 0 \; \forall a_i \in A_i$;
2 **for** $t \in \{1, \ldots, T\}$ **do**
3 $\quad \alpha \leftarrow \lambda^t$; $\epsilon \leftarrow \mu^t$; `// update learning and exploration rates`
4 $\quad \dot{a}_i^t \leftarrow \epsilon$-greedy; `// choose (and take) action using` ϵ`-greedy`
5 $\quad f_{\dot{a}_i^t} \leftarrow$ observe travel time on $\dot{a}_i^t$;
6 $\quad r(\dot{a}_i^t) \leftarrow -(f_{\dot{a}_i^t} + \beta(f_{\dot{a}_i^t} - F_{\dot{a}_i^t}))$; `// compute the reward of` $\dot{a}_i^t$
7 $\quad Q(\dot{a}_i^t) \leftarrow (1-\alpha)Q(\dot{a}_i^t) + \alpha r(\dot{a}_i^t)$; `// update Q-value of` $\dot{a}_i^t$
8 **end**

The learning process works as follows. At every episode $t \in [1, T]$, each agent $i \in D$ chooses an action $\dot{a}_i^t \in A_i$ using an ϵ-greedy exploration strategy (line 4 of Algorithm 5.1). The exploration rate ϵ at episode t is given by $\epsilon(t) = \mu^t$. After taking the chosen action, the agent observe its travel time $f_{\dot{a}_i^t}$ (line 5 of Algorithm 5.1) and computes its reward $r(\dot{a}_i^t)$ following Equation (5.2) (line 6 of Algorithm 5.1). Note that, by computing the toll only after the agent observes its travel time, we ensure that our mechanism charges tolls a posteriori. Finally, the agent updates $Q(\dot{a}_i^t)$ as in Equation (2.7) (line 7 of Algorithm 5.1). The learning rate α at episode t is given by $\alpha(t) = \lambda^t$.

Again, we remark that Algorithm 5.1 represents just an abstract scheme of the algorithm from the agent's perspective. In order to effectively run our approach, one needs firstly to load the problem instance and initialise the agents. Afterwards, for every episode (up to episode T), we can run once the main loop of each agent (i.e., where an agent chooses an action, observes the reward, and updates its Q-table). The time and space complexity of our approach is presented in the next proposition. The reader is referred to to Appendix A for the proof and for complete details on the simulation procedure.

Proposition 5.2. *Our toll-based Q-learning approach has $O(T(dK + ld + lmK))$ time complexity and $O(dK)$ space complexity, for T episodes, d drivers, and K actions, l links and m OD pairs.*

5.3 Theoretical analysis

In this section, we provide a theoretical analysis of our approach. The main objective here is to show that our approach converges to a system-efficient equilibrium (i.e., the SO), as formulated in Theorem 5.1, adapted from Beckmann, McGuire and Winsten (1956) and Roughgarden and Tardos (2002).

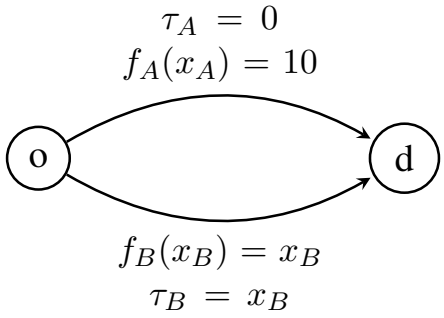
Theorem 5.1 (Beckmann, McGuire and Winsten (1956))**.** *Consider a toll-based instance $P' = (G, D, f, \tau)$ of the route choice problem, where driver $i \in D$ experiences a cost $c_l = f_l + \tau_l$ after traversing link l, with f_l and τ_l representing the travel time and toll charged at that link, respectively. Under these settings, the average travel time under UE for P' corresponds to that of the SO for $P = (G, D, f)$.*

Intuitively, Theorem 5.1 says that given an instance P of the route choice problem, if we apply MCT to it (thus resulting in an instance P' of the toll-based route choice problem), then the UE in P' will be equivalent to the SO in P. In other words, the UE with MCT achieves the same average travel time of the SO of the original problem. We refer the reader to Beckmann, McGuire and Winsten (1956) for the complete proofs. An illustrative example on how this theorem applies to Pigou (1920)'s network is presented in Example 5.1.

Example 5.1. *Consider the network in Figure 5.1, adapted from Pigou (1920), which is traversed by 10 agents. To traverse the network, each agent must take one out of two possible routes, A and B, whose travel times are given by $f_A(x_A) = 10.0$ and $f_B(x_B) = x_B$, respectively. By definition, the UE in this network is achieved when all vehicles choose route B, which results in an average travel time of 10.0. The SO, on the other hand, corresponds to the case where each route receives half of the flow, which results in an average travel time of 7.5. Here, the PoA is 4/3. Now consider the same example, but adopting the MCT scheme. The cost on each link now corresponds to the sum of its travel time (as before) and the toll charged on it, i.e., $c_l = f_l + \tau_l$. Specifically, for routes A and B we have that $c_A = 10.0 + 0.0 = 10.0$ and $c_B = x_B + x_B = 2x_B$, respectively. In this case, the UE is achieved when each route receives half of the drivers, which corresponds to an average cost of 10.0 and an average travel time of 7.5. This is precisely the SO. Hence, under MCT, we have that SO=UE and that PoA is 1.*

Note that Theorem 5.1 is about the equivalence of SO and UE under MCT. However, *it does not consider how the UE can be achieved.* In other words, Theorem 5.1 simply assumes that the UE is given. Indeed, this is a common assumption of other works in the literature, such as in Sharon et al. (2017). However, since route choice is a multiagent problem, guaranteeing convergence to the UE is not trivial (as discussed in Section 2.2). Hence, in order for Theorem 5.1 to apply to our approach, we need first to show that our approach indeed achieves the UE. In contrast to other works in the literature, we show that our method *converges to the UE*, and then we show that such UE is aligned to the SO. This is shown in Theorem 5.2. The complete proof is presented in the next subsection.

Figure 5.1: Two-route example network adapted from Pigou (1920), with travel times and toll values shown next to the corresponding routes.



Theorem 5.2. *Consider an instance P of the route choice problem. If all drivers use the Q-learning algorithm with learning rate $\alpha(t) = \lambda^t$ and exploration rate $\epsilon(t) = \mu^t$, then the system converges to the UE in the limit.*

From Theorem 5.2, we can conclude that our algorithm can find the UE both in the original problem (P) as well as in the corresponding toll-based version (P'). This means that, by employing MCT, our algorithm achieves a system-efficient equilibrium (Theorem 5.1). In other words, our approach reduces the PoA to its best ratio of 1. Therefore, based on Theorems 5.1 and 5.2 we can formulate the following corollary.

Corollary 5.2. *Consider an instance P of the route choice problem, where all drivers use the Q-learning algorithm with learning rate $\alpha(t) = \lambda^t$ and exploration rate $\epsilon(t) = \mu^t$. By employing marginal-cost tolling, the agents converge to a system-efficient equilibrium in the limit, i.e., the average travel time under UE corresponds to that of the SO. Thus, the price of anarchy converges to 1 in the limit.*

5.3.1 Convergence to the user equilibrium

In this section, we prove Theorem 5.2 by showing that our approach converges to the UE. Hereafter, by *our approach* we mean the settings presented in Section 5.2, i.e., a stateless MDP with Q-learning agents using ϵ-greedy exploration, where $\alpha(t) = \lambda^t$ and $\epsilon(t) = \mu^t$. For simplicity and without loss of generality, we assume that the actions' rewards are in the interval $[0, 1]$.

The intuition underlying the proof of Theorem 5.2 is that, given that learning (α) and exploration (ϵ) rates are decreasing with time (using decays λ and μ, respectively), then the system is becoming more stable (Theorem 5.3). We say that the environment is stabilising if randomness (due to agents exploration) is decreasing along time. Conse-

quently, we can show that, in the limit, the actions with the highest Q-values are precisely the optimal ones (Lemma 5.3), which leads the agents to exploit only optimal actions in the limit (Lemma 5.2), thus achieving the UE (Theorem 5.2).

Initially, we restate here Proposition 3.2, which defines the probability that best[1] and non-best actions are chosen by a given agent i at episode t.

Proposition 3.2. *Using ϵ-greedy exploration with $\epsilon(t) = \mu^t$, at episode t agent i chooses its best action $\overset{+}{a}{}_i^t = \arg\max_{a_i^t \in A_i} Q(a_i^t)$ with probability $\rho(\overset{+}{a}{}_i^t) = 1 - \frac{\mu^t(K-1)}{K}$ and any other action $\bar{a}_i^t \in A_i \setminus \overset{+}{a}{}_i^t$ with probability $\rho(\bar{a}_i^t) = \frac{\mu^t(K-1)}{K}$.*

From Proposition 3.2, we remark that $\overset{+}{\rho} \to 1$ and $\bar{\rho} \to 0$ as $t \to \infty$ and $\epsilon \to 0$. To this respect, as time goes to infinity, the values of α and ϵ become so small that the probability of noisy observations changing the Q-table (and, mainly, the best action) goes to zero. When the system behaves in this way, we say it is *stabilising*. Under such circumstances, we can apply Theorem 5.3, adapted from Theorems 3.1 and 3.2.

Theorem 5.3 (adapted from Theorems 3.1 and 3.2)**.** *The environment is stabilising as $t \to \infty$. In this scenario, the probability that the Q-values of best actions (of any agent) become non-best after ∇ agents decide to explore a non-best action is bounded by $O(\bar{\rho}^{\nabla}(\overset{+}{\rho} + \bar{\rho}))$, which goes to zero as $t \to \infty$.*

Observe that an agent can, eventually, change its best action given that it *is* learning. However, the agent should be able to prevent its Q-values from reflecting unrealistic observations. Of course, stability does *not* imply that the Q-value estimates are correct and that the agents are under UE. These are shown to be true, however, in Lemma 5.3 and Theorem 5.2, respectively.

We can now advance to show that, in the limit, the action with highest estimated Q-value is indeed the optimal action. To this regard, we firstly characterise the agent's behaviour in terms of the UE, as shown in the next lemma.

Lemma 5.1. *Under UE, every agent $i \in D$ using ϵ-greedy exploration exploits its best route $\overset{+}{a}_i = \arg\max_{a_i \in A_i} Q(a_i)$.*

Proof. By definition, under UE, for each pair of routes a' and a'' of the same OD pair, with $x_{a'} > 0$, we have that $r(a') \geq r(a'')$. For the sake of contradiction, assume that the system is under UE and that there exists a pair of routes a' and a'' belonging to the

[1]Hereafter, we refer to the action with highest Q-value as the *best action* and to the other actions as *non-best*. Observe that the best action is not necessarily optimal.

same OD pair for which $x_{a'} > 0$ but $r(a') < r(a'')$. Recall that we model the problem as a stateless MDP and agents as Q-learners with ϵ-greedy exploration. Consequently, Q-values can be seen as estimates of the reward values of their corresponding actions. Therefore, given that the reward on a' is lower than on a'', then all the $x_{a'}$ vehicles using a' would deviate to a'' (i.e., they would exploit a'', not a') as soon as their Q-values are correct (which is the case in the limit, as shown next in Lemma 5.3). This contradicts the initial assumption that the system is under UE, which completes the proof. □

Observe that, in the UE definition, the notion of *best* refers to the value associated with each action (route). In RL-settings, these values correspond to actions' Q-values. Therefore, now we need to show that agents actually choose actions with highest estimated Q-values and that such actions are indeed the optimal ones. These are shown in Lemmas 5.2 and 5.3, respectively.

Lemma 5.2. *In the limit, agents exploit their knowledge most of the time, i.e., they tend to choose the actions with highest estimated Q-values.*

Proof. It follows from Proposition 3.2 and Theorem 5.3, since $\overset{+}{\rho}{}_i^t \to 1$ and $\bar{\rho}_i^t \to 0$ as $t \to \infty$ and $\epsilon \to 0$. □

Lemma 5.3. *In the limit, the action with highest estimated Q-value* $\overset{+}{a}_i = \arg\max_{a_i \in A_i} Q(a_i)$ *is indeed the optimal action* $\overset{*}{a}_i = \arg\max_{a_i \in A_i} r(a_i)$*, i.e.,* $\overset{+}{a}_i = \overset{*}{a}_i$ *as* $t \to \infty$.

Proof. This lemma can be proved by contradiction. Assume that agent i has an action $\overset{+}{a}_i = \arg\max_{a_i \in A_i} Q(a_i)$ with highest estimated Q-value but that this action is not optimal, i.e., $\overset{+}{a}_i \neq \overset{*}{a}_i = \arg\max_{a_i \in A_i} r(a_i)$. In order for that be possible, we need that $r(\overset{+}{a}_i) < r(\overset{*}{a}_i)$ and $Q(\overset{+}{a}_i) > Q(\overset{*}{a}_i)$ hold at the same time. Although counter-intuitive, this behaviour often occurs in the initial episodes, given that the agents' learning process leads travel times to oscillate. In this case, some Q-values may not correspond to the most accurate reward estimate of an action. However, due to exploration, agent i will eventually take route $\overset{*}{a}_i$. Moreover, in the limit, all actions will be infinitely explored. Therefore, as $t \to \infty$, we have that $Q(\overset{*}{a}_i)$ will increase until it eventually becomes the highest one, i.e., $Q(\overset{*}{a}_i) \approx r(\overset{*}{a}_i) > Q(\overset{+}{a}_i) \approx r(\overset{+}{a}_i)$, which contradicts the initial assumption. □

We highlight that one of the key requirements of Q-learning is that each action should be infinitely explored. However, such exploration should not lead optimal actions to seem sub-optimal. This is shown in the next lemma.

Lemma 5.4. *In the limit, agents using ϵ-greedy exploration with $\epsilon(t) = \mu^t$ can still explore non-best actions without invalidating the UE, i.e., agents' exploration does not destabilise the UE.*

Proof. Suppose the system has converged to the UE in the limit (after a sufficiently large number of episodes). At this point, all agents are using their best actions, i.e., the ones with highest estimated Q-values (Lemmas 5.1 and 5.2). Observe that agents can still explore other actions, though less frequently (Proposition 3.2 and Lemma 5.2). Thus, in order to prove this lemma, one needs to show that, under UE, exploration will not generate an *abrupt* change in the Q-values. An abrupt change occurs in an agent's Q-table only if it receives a reward that leads the Q-value of a non-best action to become better than that of the best one. However, from Theorem 5.3, we have that such abrupt changes will not affect the UE and that even if they do, a little amount of additional exploration is enough to lead the Q-values back to their true values (Lemma 5.3). □

We now have the required tools for proving Theorem 5.2. Recall that our final objective is to show that our approach converges to a system-efficient equilibria (i.e., the SO) as soon as MCT is employed. From Theorem 5.1, this is only attainable if our approach is guaranteed to converge to the UE. Therefore, proving Theorem 5.2 is sufficient to show that, by employing MCT, our approach converges to the SO.

Proof of Theorem 5.2. According to Theorem 5.3, the system becomes stable in the limit and abrupt changes do not affect the Q-values (i.e., non-best actions cannot become the best ones). Moreover, from Lemma 5.2, we know that in the limit all agents keep exploiting most of the time. Remember that exploiting means choosing the action with the highest estimated Q-value, which in the limit corresponds to the optimal one, according to Lemma 5.3. Finally, from Lemma 5.4 we have that exploration does not affect the UE. Therefore, our algorithm can be said to converge to the UE. □

5.3.2 Fairness

In this section, we analyse the fairness of our approach. We begin with a more precise definition of fairness, which is given as follows.

Definition 5.1 (MCT fairness)**.** *A marginal-cost tolling scheme is fair if the agents are charged exactly their marginal costs (i.e., the cost they impose on others).*

Observe that tolls can be seen as a mean to penalise undesired (i.e., selfish) behaviour. In this sense, from Definition 5.1, we can conclude that if toll values do not correspond to marginal costs, then such tolls may end up penalising the wrong agents (i.e., those that are not acting selfishly). In other words, unfair tolling should be avoided.

In contrast to other works in the literature, our approach charges tolls a posteriori. The next theorem shows that charging agents a posteriori translates into a fairer tolling scheme, since agents only pay for the cost they are actually imposing on others. A more concrete example comparing a priori and a posteriori tolling schemes in terms of fairness is presented forward, in Example 5.2.

Theorem 5.4. *Consider a toll-based instance $P = (G, D, f, \tau)$ of the route choice problem. Then, charging tolls in P a posteriori is fairer than charging a priori.*

Proof. Building upon Definition 5.1, to show that a posteriori toll charging is fairer than a priori toll charging, we need to show that the former charges exactly the marginal cost, whereas the latter may not. For simplicity, we perform this analysis from the links perspective (although it easily extends to routes).

In general terms, the toll charged on link l is given by $\tau_l = \beta(p_1 x_l^{\beta})$ (as formulated in Proposition 5.1). Assume, without loss of generality, that $p_1 = \beta = 1$. In this case, we have that $\tau_l = x_l$, which corresponds to one of the cost functions presented in Pigou (1920)'s example. Abusing notation, assume that $\tau_l^t = x_l^t$ corresponds to the toll charged on link l at episode t based on the flow on that link at that episode. Observe that the flow on link l can change from one episode to another. This is especially true at the beginning of the learning process, when the system is not yet stable. Such a difference can be expressed as $\Delta_l^t = |x_l^{t-1} - x_l^t| \geq 0$.

In the case of *a priori* toll charging, τ_l^t is computed based on previous steps. For simplicity, assume that $\tau_l^t = x_l^{t-1}$. On the one hand, if $\Delta_l^t = 0$, then the toll τ_l^t charged on link l is precisely x_l^t, given that $x_l^{t-1} = x_l^t$. On the other hand, if the flow on link l changes from one episode to another, then $x_l^{t-1} \neq x_l^t$ and $\Delta_l^t > 0$. Observe that the marginal cost for taking link l at episode t should be x_l^t, whereas a priori toll charging considers x_l^{t-1}. Therefore, whenever $\Delta_l^t > 0$, agents using l would be charged above (or below) the cost they are actually imposing on others. Consequently, a priori toll charging is unfair whenever $\Delta_l^t > 0$. This cost can be even higher when τ_l^t is not based on the flow of a *single* previous episodes, but on *many* previous episode (e.g., an average of previous flows).

In contrast, *a posteriori* toll charging defines that $\tau_l^t = x_l^t$, which corresponds

precisely to the cost agents are imposing on others. Observe that Δ_i^t does not affect the toll values here. Thus, a posteriori toll charging (as used in our approach) can be said fairer than a priori toll charging. □

Example 5.2. *Consider again the 10-agent network presented in Example 5.1 and Figure 5.1. In this extended example, we consider a hypothetical sequence of three episodes (in which every agent chooses a route). Such a sequence is presented in Table 5.1. In the table, we present the toll values for both routes (A and B) as generated by a priori (as usual in the literature, assuming that tolls are initialised with zero, as in Sharon et al. (2017)) and a posteriori (as in our approach) tolling schemes. In the case of **a priori** tolling, assume that toll values are initialised with 0.0, as in Sharon et al. (2017). On subsequent episodes, the toll of each route is defined as the marginal cost of such route in the previous episode. The rationale behind such model is that agents can check the tolls that they are going to pay on each route before they actually take any route. However, this leads to outdated toll values. We note that, by definition, MCT schemes should charge each agent according to its marginal cost, which is not achieved by a priori tolling schemes. As seen in Table 5.1, in the second episode, even though all agents are using route B, the toll they are going to pay is only 6.0, which corresponds to 60% of their actual marginal cost. Later on, in the third episode, half of the agents are using each route, which corresponds to the SO. Nevertheless, agents using route B need to pay a toll of 10.0. Therefore, the prices charged by a priori tolling may be (and often are, as shown in this example) unfair. In the case of **a posteriori** tolling schemes, by contrast, tolls are charged only after a route is taken. At this point, one could argue that our approach prevents agents from analysing the costs of their decisions a priori. However, as tolls are incorporated into agents' utility functions, the effects of such a posteriori charges are naturally captured by the learned Q-functions. As seen in Table 5.1, the tolls defined by a posteriori tolling schemes always correspond to the actual flow of vehicles (and their marginal costs). Consequently, a posteriori tolling schemes can be said to be fairer than a priori tolling schemes.*

5.4 Experimental evaluation

In this section, we provide an empirical analysis on the performance of our approach to validate the theoretical results from previous section. Again, recall that *learn-*

Table 5.1: Comparison of a priori and a posteriori toll charging in the road network of Figure 5.1, with three timesteps.

	flow		**a priori tolling**		**a posteriori tolling**	
episode	x_A	x_B	τ_A	τ_B	τ_A	τ_B
1	4	6	0.0	0.0	0.0	6.0
2	0	10	0.0	6.0	0.0	10.0
3	5	5	0.0	10.0	0.0	5.0

ing here translates into finding the best routes to take, a target that changes steadily due to the presence of multiple agents with possibly conflicting interests. In this sense, the term convergence refers to a point where agents keep *exploiting* their knowledge (encoded by means of their Q-tables) most of the time and the system is *stable* (so that agents only observe small fluctuations in their costs). Our aim is to show that, by using our approach, such a stable point corresponds to a system-efficient equilibrium (i.e., the SO).

5.4.1 Methodology

We empirically validate our theoretical results by simulating our approach as described in Appendix A. We employ the same road networks[2] used in the previous chapters. The reader is referred to Section 3.4.1 for the detailed description of each such network. To enhance presentation, here we replicate the table summarising these networks, but now also showing the SO solution of each, as presented in Table 5.2. The most representative such networks are illustrated in Appendix B.

Following previous chapters, we limit the number of available routes to the K shortest ones in order to avoid arbitrarily large sets of routes. The set of K shortest routes of each OD pair was computed using the KSP algorithm (YEN, 1971), where the best value for K varies among the different networks, depending on their characteristics.

As usual, an experiment corresponds to a complete execution, with T =1,000 episodes (except for the SF network, in which case we set T =10,000 episodes), of our method on a single network. After an execution is completed, we measure (considering the last episode) its performance by means of the average travel time (*avg-tt* hereafter, measured in minutes),and the average *proximity to the SO*. The latter is formulated as

$$\text{proximity}(x, x^*) = 1 - \frac{|x^* - x|}{x^*}, \tag{5.3}$$

[2]The road networks are available at <https://github.com/maslab-ufrgs/network-files>.

Table 5.2: Characteristics of the networks used for validation of our approach.

Network	Nodes	Links	OD pairs	Total demand	*avg-tt*[a] under SO
B^1	4	5	1	4,200	15.00
B^2	6	9	1	4,200	≈23.33
B^3	8	13	1	4,200	32.50
B^4	10	17	1	4,200	42.00
B^5	12	21	1	4,200	≈51.66
B^6	14	25	1	4,200	≈61.43
B^7	16	29	1	4,200	71.25
BB^1	8	8	2	4,200	7.50
BB^3	12	16	2	4,200	19.00
BB^5	16	24	2	4,200	47.00
BB^7	20	32	2	4,200	120.50
OW	13	48	4	1,700	66.92
SF	24	76	528	360,600	19.95

[a] Values reported in the literature (STEFANELLO; SILVA; BAZZAN, 2016; STEFANELLO; BAZZAN, 2016).

and refers to how close the average travel time (say, x) obtained by the agents is to that of the SO (say, x^*; as reported in Table 5.2); the higher the value is to 1.0, the better.

We tested different value for the Q-learning's parameters. The tuning process is described in detail in Section 5.4.2. The results of the best configurations were then selected for further analyses in Section 5.4.3.

The algorithms, data analysis and plots were all implemented in Python 2.7.

5.4.2 Parameter tuning

Our toll-based Q-learning approach has three parameters: K (number of routes), λ (decay rate of α) and μ (decay rate of ϵ). In order to better assess our approach, we tested different values for these parameters. For the number of routes, we used $K \in \{4, 8, 12, 16\}$. After extensive tests, we found that intermediate values for K pose no significant difference on the results. In the case of the decay rates, we tested the values $\{0.98, 0.99, 0.995, 0.999\}$, with $\lambda = \mu$. Recall that the learning and exploration rates, α and ϵ, are initialised with 1.0 and multiplied by their decay rates after each episode. Thus, lower decay rates were not used to ensure the agents keep learning/exploring for a longer time. As for the SF network, considering the high number of vehicles using it (i.e., two orders of magnitude higher than in the other networks), we tested $\lambda \in \{0.9995, 0.9997, 0.9999\}$ and $\mu \in \{0.995, 0.997, 0.999\}$. Observe that, in

Table 5.3: Parameters' configuration that produced the best results for each network.

Network	K	λ	μ
B^1	3	0.99	0.99
B^2	5	0.99	0.99
B^3	7	0.99	0.99
B^4	9	0.99	0.99
B^5	11	0.99	0.99
B^6	13	0.99	0.99
B^7	15	0.99	0.99
BB^1	3	0.98	0.98
BB^3	8	0.99	0.99
BB^5	4	0.99	0.99
BB^7	4	0.99	0.99
OW	8	0.99	0.99
SF	10	0.9997	0.999

the SF network, the values used for λ were higher than for μ to ensure that agents keep learning even after exploration is decreased. We ran 30 repetitions for each combination of values for these parameters and compared such combinations using Student's t-tests at the 5% significance level, except if otherwise stated. The best performing combination of parameters for each network is presented in Table 5.3.

As seen in Table 5.3, in general, the larger the network, the higher the value required for K. This is because the number of possible routes increases with the size of the network. Consequently, a higher value for K is necessary to efficiently spread the traffic on these networks. Observe that, as compared to our regret-minimising approach, here the Braess networks depend more strongly on a high number of routes to converge properly. The rationale here is that these networks have two kinds of routes: those with and those without the *zero-cost link* (the one that causes the Braess paradox). The routes of the former kind are the shortest ones, but they are more sensitive to congestions and, thus, have higher marginal-costs. Consequently, agents can only spread efficiently across these networks if the low marginal-cost routes are present as well. This explains the need for higher values for K. Interestingly, this was not a problem in the case of our regret-minimising approach (Chapters 3 and 4) because, without tolls, drivers focus on minimising their travel times, thus rendering non-shortest routes unnecessary. Also considering the number of routes, observe that the bi-commodity version of the Braess graphs required fewer routes (in spite of their larger size) as compared to the single-commodity ones. The point is that, in these networks, the routes with the zero-cost link are much less sensitive to congestions, thus having a lower marginal cost. We refer the reader to Ste-

Table 5.4: Average performance (with standard deviation in parentheses) of our approach (Ours) on different networks in terms of proximity to the SO.

Network	Proximity to the SO
B^1	0.9999 (10^{-6})
B^2	0.9998 (10^{-5})
B^3	0.9999 (10^{-5})
B^4	0.9999 (10^{-5})
B^5	0.9999 (10^{-5})
B^6	0.9999 (10^{-5})
B^7	0.9999 (10^{-5})
BB^1	1.0000 (0.000)
BB^3	0.9999 (10^{-4})
BB^5	0.9999 (10^{-5})
BB^7	0.9996 (10^{-4})
OW	0.9990 (10^{-5})
SF	0.9954 (10^{-4})
Average	0.9995 (10^{-3})

fanello and Bazzan (2016) for a more detailed discussion on the particular characteristics of the Braess graphs.

Regarding the decay rates, in general, the same values achieved the best results in all networks. This is a consequence of networks' size and demand (number of agents). In general, as discussed in previous chapters, the larger the number of agents, the longer it takes for them to learn their best routes. As a result, the values of λ and μ need to be higher (which translates into slower decays) to ensure that agents explore their routes sufficiently.

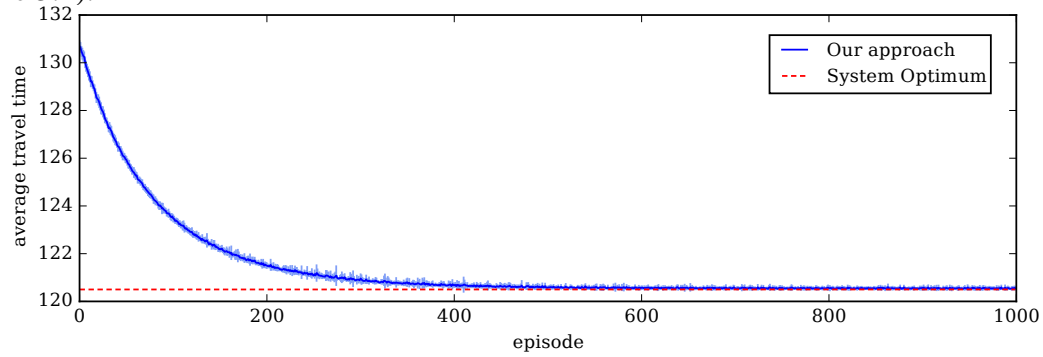
The results of the above best configurations were selected for further analyses in the next subsection.

5.4.3 Results

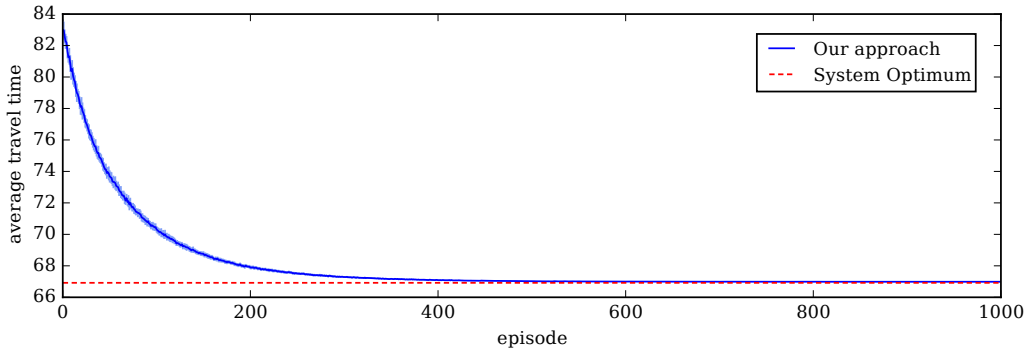
The average performance of our approach in different networks in terms of proximity to the SO is presented in Table 5.4. Results represent averages over 30 repetitions, with standard deviations shown in parentheses. The values of the algorithms' parameters are those listed in Table 3.2.

As seen in the table, our approach obtains good approximations of the SO in the tested networks. On average, our results are within 99.95% of the SO, with a standard deviation of 0.13%. We remark that this resulting solution concept corresponds either to

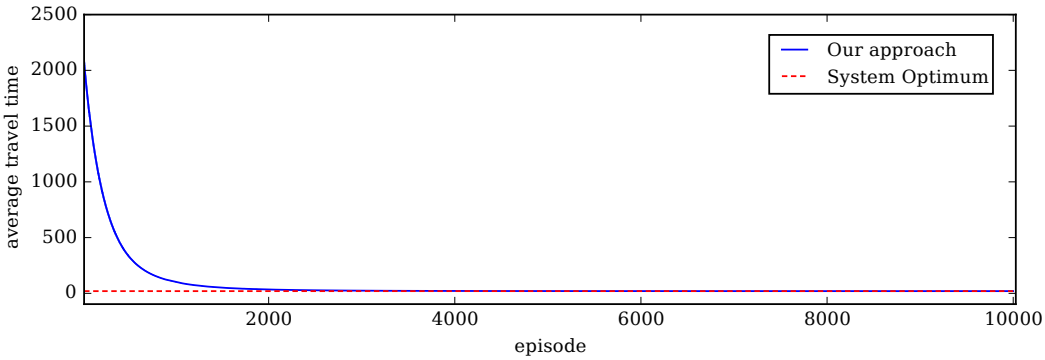
Figure 5.2: Average travel time along episodes in selected networks, with shaded lines representing the standard deviation and dashed lines representing the SO (as reported in Table 5.2).



(a) BB^7 network



(b) OW network



(c) SF network

the UE and to the SO. In other words, the average travel times achieved here are minimum and, due to the toll values, no agent have an incentive to deviate. Therefore, as expected, the experimental results are consistent with the theoretical analysis presented in Section 5.3, showing that our approach converges to a system-efficient equilibrium.

In order to better analyse the agents' learning behaviour, Figure 5.2 presents the average travel time along episodes in selected networks. In the plots, each curve is an average over 30 repetitions (with standard deviation shown as shaded lines) and dashed lines present the average travel times under SO (as reported in Table 5.2). We remark that the SF network was run for more episodes than the other networks, which justifies the

high concentration in the left side of the plot in Figure 5.2c.

The plots show that, as for our regret-minimising approaches (Chapters 3 and 4), the average travel time here is high in the beginning as a consequence of the Q-table initialisation. In spite of the presence of the tolls, however, we can see that agents were able to learn their best routes reasonably fast as they become more experienced. Note that, as compared to our regret-minimising approaches, the average travel time here is is much higher in the first episodes. This is a direct consequence of the tolls. In the very beginning, agents that take shortest routes fare better than the others. However, the other agents start to prefer such routes as well, thus increasing their marginal costs (i.e., the peak in Figure 5.2c is higher than 2,000, whereas in Figure 3.2c it is lower than 700). As a consequence, the shortest routes rapidly become unattractive to all agents. Such a phenomenon is particularly evident when the number of agents is higher, as is the case of the SF network.

Thus, our results are consistent with the theoretical analysis presented in Section 5.3, which confirms our initial hypothesis, namely that our reinforcement learning, MCT-based approach converges to the SO.

5.5 Related work

In this section we discuss representative literature on system-efficient equilibria in route choice (and related problems). The reader is referred to Essen et al. (2016) and Ortúzar and Willumsen (2011) for a more detailed overview.

The use of tolls to enforce system-efficient behaviour has been widely explored in the literature. There is a plethora of works in this line, considering drivers with heterogeneous utility (COLE; DODIS; ROUGHGARDEN, 2003), toll information mechanisms (KOBAYASHI; DO, 2005), tolls with bounded values (BONIFACI; SALEK; SCHÄFER, 2011), RL-based tolls (TAVARES; BAZZAN, 2014), and so on. We concentrate, however, in the MCT scheme (PIGOU, 1920). The concept of MCT has been investigated in several works, such as Sharon et al. (2017), Ye, Yang and Tan (2015), Yang, Meng and Lee (2004), and Meir and Parkes (2016). As opposed to our approach, nonetheless, these works neither investigate how drivers react to tolls nor ensure convergence to the UE (which, by using MCT, is then aligned to the SO). Furthermore, these tolling schemes charge tolls *a priori*, i.e., before the agents start their trips. Ideally, however, tolls should only be charged after their real marginal costs are available (i.e., at the end of the trips).

A priori tolling is indeed appealing from the agents' perspective, since such agents can see in advance the toll associated with each of their possible actions. Nonetheless, these schemes usually define the prices based on historical congestion levels, meaning that the agents may end up paying a toll that is higher than their marginal costs. In particular, since MCT is based on the impact an agent causes on others, one cannot assess such impact before it happens (except if one can predict drivers decisions along their trips). Hence, we say that these schemes are unfair, as discussed in Section 5.3.2.

In this work, by contrast, we assumed that tolls are charged *a posteriori* and *per route*. We then presented a general toll formulation that can be computed directly by the agents. In this way, we can simplify the infrastructure requirements for deploying the tolling scheme by assuming that each vehicle has a navigation device responsible for charging the toll whenever a trip is finished. As reported by the National Surface Transportation Infrastructure Financing Commission (2009), this makes the agents' decision process easier since the drivers can better understand the costs they are being charged. Traditional tolling schemes could also benefit from connected navigation devices. However, such approaches would strongly depend on stable communication (otherwise tolls would not be available a priori), whereas our approach remains robust even under precarious communication conditions (since tolls could be computed at any time after each trip is finished).

Other works investigate the SO by explicitly assuming that agents behave altruistically. Chen and Kempe (2008) and Hoefer and Skopalik (2009) investigated routing games assuming that agents can present altruistic behaviour. In these approaches, an agent's perceived cost is a linear combination of its travel time (selfish objective) and of how much its decision deteriorates others' travel times (altruistic objective). This formulation is equivalent to MCT, except for the fact that the contribution of each term (selfishness/altruism) is controlled by a parameter β, which is known as the altruism level. Altruism was also the focus of Levy and Ben-Elia (2016)'s work, which employed an agent-based model where drivers choose their routes based on subjective estimates over their costs. According to Fehr and Fischbacher (2003), under certain conditions, real drivers are indeed willing to take altruistic decisions so as to improve the global performance. However, the higher the price of such a social behaviour, the less frequent it is. Therefore, altruism cannot be explicitly imposed on the agents. Furthermore, these works assume that agents know each others' payoff to compute their utilities.

The use of route guidance mechanisms to bias drivers' decisions towards the SO

has also been approached in the literature. A good review on the topic is provided by Essen et al. (2016). Lujak, Giordani and Ossowski (2015) proposed a negotiation mechanism where several types of agents negotiate the traffic assignment as a whole. However, they assume that additional infrastructure (e.g., agents to control origins and intersections) exist. Bazzan and Klügl (2005) investigated the impact of providing biased recommendations to drivers. Specifically, a centralised service suggests routes aiming at a system-efficient equilibrium. Notwithstanding, biasing the provided suggestions may lead to unfair assignments. Moreover, in general, these works assume that a centralised mechanism makes such biased suggestions to the drivers. As discussed by Jahn et al. (2005), some drivers are willing to bear the cost of socially desired routes (up to certain limits) if the traffic system suggests them to do so. However, experiments with human subjects evidence the adoption of such mechanisms is low (ESSEN et al., 2016; RIETVELD, 2010).

Wolpert and Tumer (1999, 2002) introduced the idea of *difference rewards*, which also relates to our approach. Basically, the difference reward an agent receives for taking an action corresponds to the amount the system's performance deteriorates considering his action. Precisely, it is measured as the difference between the system's performance with and without it. Using difference rewards, the agents' reward signal is aligned with the system's utility so that they converge to the SO. However, difference rewards can only be computed upon strong, full observability assumptions. Later on, methods for approximating the difference reward signals were proposed, as in the work of Agogino and Tumer (2004), for instance. However, this kind of approach still depend on some sort of global information.

Christodoulou, Mehlhorn and Pyrga (2014) introduced a coordination mechanism through which links' cost functions are adjusted to account for agents selfish behaviour. Specifically, cost functions remain unchanged up to a flow threshold, after which the cost is set to infinity. The idea is to bias agents' decisions towards the SO. Nonetheless, the cost functions must be defined by a central authority. Additionally, that method is only applicable to simple, parallel links scenarios.

Finding system-efficient equilibria has also been approached with metaheuristics. Dias et al. (2014) employed an ant colony optimization algorithm in which vehicles deposit pheromone into the roads they travel to repel other vehicles. A genetic algorithm was used by Cagara, Bazzan and Scheuermann (2014) to optimise the assignment of vehicles. Buriol et al. (2010) considered the toll booth problem, in which one aims at selecting a subset of links to charge tolls, and employed a biased random-key genetic algorithm.

However, these approaches rely on centralised mechanisms (either for generating the assignment or for aggregating the drivers' knowledge), and the individual agents' decision processes are not modelled or taken into account.

5.6 Discussion

In this chapter, we proposed an a posteriori tolling scheme through which reinforcement learning (RL) agents are guaranteed to converge to a system efficient equilibrium (i.e., the system optimum—SO). In the route choice problem, driver-agents minimise their travel costs, which leads to the user equilibrium (UE). In order to bias the UE towards the SO, we employed the concept of marginal-cost tolling (MCT), where agents are charged proportionally to the cost they impose on others. In our approach, agents are charged a posteriori (as opposed to the literature, where agents are charged a priori), thus allowing the tolls to be computed by the agents themselves, and eliminating the need for additional infrastructure. We generalised the toll values formulation for univariate, homogeneous polynomial cost functions (which encompasses the most commonly-used cost functions in the literature). Our toll formulation allows the agents to compute the toll associated with their routes using only their own knowledge.

We provided theoretical and experimental results. Specifically, we proved that agents converge to the UE in the limit, and that such UE corresponds to the SO. As a consequence, in the limit, the price of anarchy achieved by our approach achieves its best ratio. Moreover, we have shown that, as compared to the existing literature, our MCT scheme is more general (it applies to most traffic scenarios), fairer (agents pay for their actual marginal costs), it can be computed by the agents themselves (it does no rely on a central authority), it is easier to deploy (it has fewer infrastructure requirements). On the experimental side, we validated our theoretical results in several road networks available in the literature, achieving an average proximity to the SO of 99.95%. Our results confirm our initial hypothesis, showing that our reinforcement learning, MCT-based approach converges to the SO.

6 CONCLUSIONS

In this thesis, we investigated the performance of reinforcement learning (RL) agents in the context of route choice, and delivered formal convergence guarantees. Multiagent reinforcement learning (MARL) is challenging because agents' decisions affect the utility of each other, which makes their objective a moving target. Route choice is a MARL problem that concerns how drivers behave when choosing routes so as to minimise their travel costs, and thus represents a particularly relevant scenario for MARL. The main challenge in route choice is that the route chosen by a driver may affect the travel times perceived by other drivers. We highlight that such complex dynamics represent the main challenge of MARL in general (i.e., not only in the context of route choice). In this sense, formal analyses regarding the system's performance using MARL are typically limited to specific scenarios.

The performance of route choice is commonly analysed in relation to two fundamental solution concepts: the user equilibrium (UE, where no agent benefits from unilaterally changing its route) and the system optimum (SO, which represents the system at its best operation). The main goal of this thesis is to show that MARL can be guaranteed to converge to the UE and to the SO (though not necessarily at the same time) upon certain conditions. In this regard, this thesis contributes to advancing the state-of-the-art literature on MARL in three ways.

The first contribution of this thesis consists in a regret-minimising Q-learning algorithm, through which agents are guaranteed to converge to the UE. In this front, agents employ the regret associated with their actions to guide their learning process. Roughly, an action's regret measures how bad it performs as compared to the other actions. We call this the action regret, which can be estimated by the agents using only their previous experiences. Building upon this formulation, we showed that employing such regret as reinforcement signal leads the agents to minimise their regret. Consequently, we proven that the system converges to an approximate UE. Additionally, we tested our approach in several instance of the route choice problem, thus confirming our theoretical results.

As the second contribution of this thesis, we extended the previous method to deal with non-local information. We defined a (mobile) navigation service (which we call the *app*) that provides non-local information (here, simply the routes average travel times) to the agents. The agents employ the app's information to improve the estimated regret of their actions. We validated our approach in several instances of the problem,

and empirically found that drivers' performance is enhanced when agents use the app informations to estimate their actions' regret.

The third contribution of this thesis refers to a toll-based Q-learning algorithm, through which the system is guaranteed to converge to a system-efficient equilibrium, i.e., an UE aligned to the SO. We developed a marginal-cost tolling (MCT) scheme, where the toll an agent pays on a link is proportional to the cost it imposes on others. We then generalised the toll values formulation for univariate, homogeneous polynomial cost functions, which comprises the most commonly-used cost functions in the literature. Using our formulation, tolls can be computed by the agents themselves and can be charged a posteriori (i.e., when the trip ends). We provided theoretical results, showing that our approach converges to the SO and that our tolling scheme is fairer than a priori schemes. As for the other fronts, we empirically validated our approach in several route choice instances, thus achieving results that corroborate with our theoretical analysis.

Together, the aforementioned fronts of this thesis provide an answer to our initial question (as presented in the introduction of this thesis). Specifically, we have shown that, using the proposed methods, it is possible to formally guarantee that reinforcement learning agents will converge to the UE and to the SO (though not necessarily at the same time). In this regard, at least in the context of route choice, MARL can achieve convergence guarantees.

Finally, we highlight that, although this thesis has considered the route choice problem in particular, our approach is not necessarily limited in this respect. In fact, our analyses may apply to other MARL problems as well. In principle, our regret-based analyses (Chapters 3 and 4) may be applied to any problem representable as a stateless MDP, with a finite set of agents and actions, where agents can keep the history of rewards received on previous episodes. In the case of our toll-based analyses (Chapter 5), apart of the previous characteristics, the problem should also present univariate, homogeneous polynomial cost functions. Furthermore, given that tolls are defined as a function of the marginal costs, rewards need to correspond precisely to an agent's actual action (e.g., proportional to travel time on the chosen route) not to the system's overall state (e.g., proportional to average travel time of all agents).

6.1 Future work

As future work, we outline the following research directions:

- Investigate alternative regret formulations to accelerate the learning process. As discussed in Chapter 3, when evaluating an action, our regret formulation considers the average of *all* rewards of that action (i.e., since the first episode). As an undesired effect, outdated rewards may lead an action to seem better or worse than it actually is, thus making the learning process slower. Building upon our regret formulation, a promising alternative consists in progressively discounting old rewards so as to put more weight on recent rewards. Additionally, we could design a mechanism that actively detects relevant changes in the rewards and eliminates only outdated rewards.
- Extend our results to consider mixed strategies and complement our convergence proofs with convergence *rate* analyses. In this thesis, agents' strategies are deterministic, although exploration adds a level of stochasticity to them. Using mixed strategies adds another level of complexity to the theoretical analysis, but results in stronger results. Furthermore, observe that although we prove that agents converge to the desired solution concepts, the rate at which such converge takes place is not contemplated by our analyses. As a consequence, when agents are learning, we cannot anticipate *when* convergence will be reached. Such analysis is important to better asses the kinds of problems to which our approach is suitable.
- Enhance the tolling scheme so that the agents can anticipate the routes' tolls. The tolling scheme presented in Chapter 5 allows tolls to be computed a posteriori in a distributed way. The drawback of this mechanism is that, in the worst case, agents may end up paying a cost arbitrarily higher than expected. The most straightforward approach to overcome such problem would be to keep, for each agent, an internal estimate of tolls based on previous episodes. Such estimates could then be represented probabilistically based on the exploration rate of other agents (i.e., estimates tend to be more precise when exploration is lower). Furthermore, such probabilities could be used to establish a bound on the worst possible loss associated with each route.
- Reformulate the use of app information to improve learning efficiency. The experiments of Chapter 4 shown that the app information improves agents' estimates on

their rewards. We seen that such information is particularly useful in the beginning of the learning process, when agents have not enough experience to take their decisions. In this sense, our approach could be changed to weigh app information based on the agents' uncertainty regarding each action. In this way, agents could use app information only when indeed needed, thus improving learning efficiency. Also important, we could extend our experimental analysis to investigate how the weight given to the app information correlate with performance improvements.

- Extend our algorithms and analyses to the dynamic shortest paths problem. As outlined in Section 2.1.2, this problem is more challenging than route choice because drivers do not know their possible routes a priori. Hence, agents must learn their routes by exploring the entire network. Although unrealistic from the traffic perspective, this problem can provide insightful results that could then be extended to other problems. In fact, we consider this work as our first step towards more general MARL convergence guarantees. In this regard, we also plan to extend our analyses to more general classes of MARL problems.
- Study the impact of multiple, competitive objectives. This line of research has been widely considered in the literature. Here, one investigates what happens when agents have multiple objectives that may burden each other. As a typical example, think of a driver that needs to decide between travelling faster or saving fuel. In this case, one tries to improve one objective only if it does not deteriorates the other. The set of such possible improvements delineates the Pareto frontier. Hence, extending our analyses to deal with multi-objectives represents another interesting direction.
- Investigate the effect of alternative communication models. In Chapter 4, we used an app to provide information to the agents. Nonetheless, several other alternatives can be seen as information sources. In particular, drivers can communicate with each other to exchange traffic advices. We consider this a relevant research direction because, apart of the travel times, such kind of communication could be useful for the agents to anticipate undesired traffic conditions, such as congestions close to a concert or flooded lanes after heavy rain.

REFERENCES

ABDALLAH, S.; LESSER, V. Learning the task allocation game. In: **Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS06)**. Hakodate, Japan: New York: ACM Press, 2006. p. 850–857.

ABERNETHY, J. D.; HAZAN, E.; RAKHLIN, A. Interior-point methods for full-information and bandit online learning. **IEEE Transactions on Information Theory**, v. 58, n. 7, p. 4164–4175, jul 2012. ISSN 0018-9448.

AGARWAL, A.; DEKEL, O.; XIAO, L. Optimal algorithms for online convex optimization with multi-point bandit feedback. In: KALAI, A. T.; MOHRI, M. (Ed.). **The 23rd Conference on Learning Theory**. Haifa: [s.n.], 2010. p. 28–40. ISBN 9780982252925.

AGOGINO, A. K.; TUMER, K. Unifying temporal and structural credit assignment problems. In: **Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2**. New York: IEEE Computer Society, 2004. (AAMAS '04), p. 980–987.

ARORA, R.; DEKEL, O.; TEWARI, A. Online bandit learning against an adaptive adversary: from regret to policy regret. In: LANGFORD, J.; PINEAU, J. (Ed.). **Proceedings of the 29th International Conference on Machine Learning (ICML 2012)**. Edinburgh: [s.n.], 2012. p. 1503–1510. ISBN 9781450312851.

AUER, P.; CESA-BIANCHI, N.; FISCHER, P. Finite-time analysis of the multiarmed bandit problem. **Machine Learning**, v. 47, n. 2/3, p. 235–256, 2002. ISSN 08856125.

AUER, P. et al. The nonstochastic multiarmed bandit problem. **SIAM Journal on Computing**, v. 32, n. 1, p. 48–77, 2002.

AUMANN, R. J. Subjectivity and correlation in randomized strategies. **Journal of mathematical Economics**, v. 1, n. 1, p. 67–96, 1974.

AWERBUCH, B.; KLEINBERG, R. D. Adaptive routing with end-to-end feedback: Distributed learning and geometric approaches. In: **Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing**. New York: ACM, 2004. (STOC '04), p. 45–53. ISBN 1-58113-852-0.

BAIRD, L. C. Reinforcement learning in continuous time: advantage updating. In: **International Conference on Neural Networks**. [S.l.]: IEEE, 1994. v. 4, p. 2448–2453.

BANERJEE, B.; PENG, J. Efficient no-regret multiagent learning. In: **Proceedings of the Twentieth National Conference on Artificial Intelligence**. [S.l.]: AAAI Press, 2005. p. 41–46.

BAR-GERA, H. Traffic assignment by paired alternative segments. **Transportation Research Part B: Methodological**, v. 44, n. 8-9, p. 1022–1046, sep 2010. ISSN 01912615.

BAZZAN, A. L. C.; KLÜGL, F. Case studies on the Braess paradox: simulating route recommendation and learning in abstract and microscopic models. **Transportation Research C**, v. 13, n. 4, p. 299—319, August 2005.

BAZZAN, A. L. C.; KLÜGL, F. Re-routing agents in an abstract traffic scenario. In: ZAVERUCHA, G.; COSTA, A. L. da (Ed.). **Advances in artificial intelligence**. Berlin: Springer-Verlag, 2008. (Lecture Notes in Artificial Intelligence, 5249), p. 63–72.

BAZZAN, A. L. C.; KLÜGL, F. **Introduction to Intelligent Systems in Traffic and Transportation**. [S.l.]: Morgan and Claypool, 2013. 1-137 p. (Synthesis Lectures on Artificial Intelligence and Machine Learning, 3).

BECKMANN, M.; MCGUIRE, C. B.; WINSTEN, C. B. **Studies in the Economics of Transportation**. New Haven: Yale University Press, 1956.

BELL, D. E. Regret in decision making under uncertainty. **Operations Research**, v. 30, n. 5, p. 961–981, 1982.

BEN-ELIA, E. et al. The impact of travel information's accuracy on route-choice. **Transportation Research Part C: Emerging Technologies**, v. 26, p. 146–159, Jan 2013.

BEN-ELIA, E.; ISHAQ, R.; SHIFTAN, Y. "If only I had taken the other road...": regret, risk and reinforced learning in informed route-choice. **Transportation**, v. 40, n. 2, p. 269–293, Feb 2013.

BLUM, A.; EVEN-DAR, E.; LIGETT, K. Routing without regret: On convergence to nash equilibria of regret-minimizing algorithms in routing games. **Theory of Computing**, v. 6, n. 1, p. 179–199, 2010. ISSN 1557-2862.

BLUM, A.; MANSOUR, Y. Learning, regret minimization, and equilibria. In: NISAN, N. et al. (Ed.). **Algorithmic game theory**. [S.l.]: Cambridge University Press, 2007. p. 79–102. ISBN 9780521872829.

BONIFACI, V.; SALEK, M.; SCHÄFER, G. Efficiency of restricted tolls in non-atomic network routing games. In: PERSIANO, G. (Ed.). **Algorithmic Game Theory: Proceedings of the 4th International Symposium (SAGT 2011)**. Amalfi: Springer Berlin Heidelberg, 2011. p. 302–313. ISBN 978-3-642-24829-0.

BOWLING, M. Convergence and no-regret in multiagent learning. In: SAUL, L. K.; WEISS, Y.; BOTTOU, L. (Ed.). **Advances in Neural Information Processing Systems 17: Proceedings of the 2004 Conference**. [S.l.]: MIT Press, 2005. p. 209–216.

BRAESS, D. Über ein Paradoxon aus der Verkehrsplanung. **Unternehmensforschung**, v. 12, p. 258, 1968.

BUREAU OF PUBLIC ROADS. **Traffic Assignment Manual**. Washington, D.C., 1964.

BURIOL, L. S. et al. A biased random-key genetic algorithm for road congestion minimization. **Optimization Letters**, v. 4, p. 619–633, 2010.

BUŞONIU, L.; BABUSKA, R.; SCHUTTER, B. D. A comprehensive survey of multiagent reinforcement learning. **Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on**, IEEE, v. 38, n. 2, p. 156–172, 2008.

CAGARA, D.; BAZZAN, A. L. C.; SCHEUERMANN, B. Getting you faster to work: A genetic algorithm approach to the traffic assignment problem. In: **Proceedings of the 16th Annual Conference on Genetic and Evolutionary Computation (companion)**. ACM, 2014. (GECCO '14), p. 105–106.

CENTRE FOR ECONOMICS AND BUSINESS RESEARCH. **The future economic and environmental costs of gridlock in 2030**: An assessment of the direct and indirect economic and environmental costs of idling in road traffic congestion to households in the UK, France, Germany and the USA. London, 2014. Available from Internet: <https://trid.trb.org/view/1329874>. Accessed March 15, 2018.

CESA-BIANCHI, N.; LUGOSI, G. **Prediction, Learning, and Games**. Cambridge: Cambridge University Press, 2006. ISSN 0040-1706. ISBN 9780511546921.

CHAN, H.; JIANG, A. X. Congestion games with polytopal strategy spaces. In: KAMBHAMPATI, S. (Ed.). **Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence**. New York: AAAI Press, 2016. p. 165–171.

CHEN, P.-A.; KEMPE, D. Altruism, selfishness, and spite in traffic routing. In: RIEDL, J.; SANDHOLM, T. (Ed.). **Proceedings of the 9th ACM conference on Electronic commerce (EC '08)**. New York: ACM Press, 2008. p. 140–149. ISBN 9781605581699.

CHIEN, S.; SINCLAIR, A. Convergence to approximate nash equilibria in congestion games. In: GABOW, H. (Ed.). **Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '07)**. New Orleans: Society for Industrial and Applied Mathematics, 2007. p. 169–178. ISBN 9780898716245.

CHORUS, C. G. A new model of random regret minimization. **European Journal of Transport and Infrastructure Research**, v. 10, n. 2, p. 181–196, Jun 2010.

CHORUS, C. G. Regret theory-based route choices and traffic equilibria. **Transportmetrica**, v. 8, n. 4, p. 291–305, 2012.

CHORUS, C. G.; ARENTZE, T. A.; TIMMERMANS, H. J. A random regret-minimization model of travel choice. **Transportation Research Part B: Methodological**, v. 42, n. 1, p. 1–18, 2008.

CHORUS, C. G.; WALKER, J. L.; BEN-AKIVA, M. A joint model of travel information acquisition and response to received messages. **Transportation Research Part C: Emerging Technologies**, v. 26, p. 61–77, 2013.

CHRISTODOULOU, G.; MEHLHORN, K.; PYRGA, E. Improving the price of anarchy for selfish routing via coordination mechanisms. **Algorithmica**, v. 69, n. 3, p. 619–640, Jul 2014. ISSN 0178-4617.

CLAUS, C.; BOUTILIER, C. The dynamics of reinforcement learning in cooperative multiagent systems. In: **Proceedings of the Fifteenth National Conference on Artificial Intelligence**. [S.l.: s.n.], 1998. p. 746–752.

COLE, R.; DODIS, Y.; ROUGHGARDEN, T. Pricing network edges for heterogeneous selfish users. In: **Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing**. New York: ACM, 2003. (STOC '03), p. 521–530. ISBN 1-58113-674-9.

DANI, V.; KAKADE, S. M.; HAYES, T. P. The price of bandit information for online optimization. In: PLATT, J. C. et al. (Ed.). **Advances in Neural Information Processing Systems 20 (NIPS 2007)**. [S.l.]: Curran Associates, Inc., 2007. p. 345–352.

DELL'ORCO, M.; MARINELLI, M. Modeling the dynamic effect of information on drivers' choice behavior in the context of an advanced traveler information system. **Transportation Research Part C: Emerging Technologies**, v. 85, p. 168–183, Sep 2017.

DIA, H.; PANWAI, S. Modelling drivers' compliance and route choice behaviour in response to travel information. **Nonlinear Dynamics**, v. 49, n. 4, p. 493–509, 2007.

DIA, H.; PANWAI, S. **Intelligent Transport Systems: Neural Agent (Neugent) Models of Driver Behaviour**. LAP Lambert Academic Publishing, 2014. ISBN 9783659528682.

DIAS, J. C. et al. An inverted ant colony optimization approach to traffic. **Engineering Applications of Artificial Intelligence**, v. 36, n. 0, p. 122–133, 2014. ISSN 0952-1976.

ESSEN, M. van et al. From user equilibrium to system optimum: a literature review on the role of travel information, bounded rationality and non-selfish behaviour at the network and individual levels. **Transport Reviews**, v. 36, n. 4, p. 527–548, jul 2016. ISSN 0144-1647.

FABRIKANT, A.; PAPADIMITRIOU, C.; TALWAR, K. The complexity of pure Nash equilibria. In: BABAI, L. (Ed.). **Proceedings of the thirty-sixth annual ACM symposium on Theory of computing - STOC '04**. Chicago, USA: ACM Press, 2004. p. 604–612.

FEHR, E.; FISCHBACHER, U. The nature of human altruism. **Nature**, v. 425, n. 6960, p. 785–791, oct 2003. ISSN 0028-0836.

FISCHER, S.; RÄCKE, H.; VÖCKING, B. Fast convergence to wardrop equilibria by adaptive sampling methods. **SIAM Journal on Computing**, v. 39, n. 8, p. 3700–3735, jan 2010. ISSN 0097-5397.

FISHBURN, P. C. Nontransitive measurable utility. **Journal of Mathematical Psychology**, v. 26, n. 1, p. 31–67, 1982.

FOSTER, D. P.; VOHRA, R. Regret in the on-line decision problem. **Games and Economic Behavior**, v. 29, n. 1, p. 7–35, 1999.

GAO, S.; FREJINGER, E.; BEN-AKIVA, M. Adaptive route choices in risky traffic networks: A prospect theory approach. **Transportation Research Part C: Emerging Technologies**, v. 18, n. 5, p. 727–740, 2010.

GRUNITZKI, R.; RAMOS, G. de O.; BAZZAN, A. L. C. Individual versus difference rewards on reinforcement learning for route choice. In: **Intelligent Systems (BRACIS), 2014 Brazilian Conference on**. [s.n.], 2014. p. 253–258. Available from Internet: <http://doi.org/10.1109/BRACIS.2014.53>. Accessed December 29, 2014.

HALL, R. Route choice and advanced traveler information systems on a capacitated and dynamic network. **Transportation Research C**, v. 4, p. 289–306, 1996.

HANNAN, J. Approximation to Bayes risk in repeated play. **Contributions to the Theory of Games**, v. 3, p. 97–139, 1957.

HART, S.; MAS-COLELL, A. A simple adaptive procedure leading to correlated equilibrium. **Econometrica**, v. 68, n. 5, p. 1127–1150, sep 2000. ISSN 0012-9682.

HASAN, M. R. et al. A multiagent solution to overcome selfish routing in transportation networks. In: **2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)**. [S.l.: s.n.], 2016. p. 1850–1855.

HE, J. et al. Improving traffic prediction with tweet semantics. In: **Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence**. AAAI Press, 2013. p. 1387–1393.

HEIDARI, H.; KEARNS, M.; ROTH, A. Tight policy regret bounds for improving and decaying bandits. In: KAMBHAMPATI, S. (Ed.). **Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence**. New York: AAAI Press, 2016. p. 1562–1570.

HENNES, D.; KAISERS, M.; TUYLS, K. RESQ-learning in stochastic games. In: **Proceedings of the AAMAS Workshop on Adaptive Learning Agents (ALA 2010)**. Toronto: [s.n.], 2010. p. 8–15.

HOEFER, M.; SKOPALIK, A. Altruism in atomic congestion games. In: FIAT, A.; SANDERS, P. (Ed.). **17th Annual European Symposium on Algorithms**. Copenhagen: Springer Berlin Heidelberg, 2009. p. 179–189. ISBN 9783642041280.

HOEFFDING, W. Probability inequalities for sums of bounded random variables. **Journal of the American Statistical Association**, v. 58, n. 301, p. 13–30, 1963.

HU, J.; WELLMAN, M. P. Multiagent reinforcement learning: Theoretical framework and an algorithm. In: **Proc. 15th International Conf. on Machine Learning**. [S.l.]: Morgan Kaufmann, 1998. p. 242–250.

HU, J.; WELLMAN, M. P. Nash q-learning for general-sum stochastic games. **Journal of Machine Learning Research**, v. 4, p. 1039–1069, 2003.

JAHN, O. et al. System-optimal routing of traffic flows with user constraints in networks with congestion. **Operations Research**, v. 53, n. 4, p. 600–616, 2005.

KAELBLING, L. P.; LITTMAN, M.; MOORE, A. Reinforcement learning: A survey. **Journal of Artificial Intelligence Research**, v. 4, p. 237–285, 1996.

KLÜGL, F.; BAZZAN, A. L. C. Route decision behaviour in a commuting scenario. **Journal of Artificial Societies and Social Simulation**, v. 7, n. 1, 2004.

KOBAYASHI, K.; DO, M. The informational impacts of congestion tolls upon route traffic demands. **Transportation Research A**, v. 39, n. 7–9, p. 651–670, August-November 2005.

LEBLANC, L. J.; MORLOK, E. K.; PIERSKALLA, W. P. An efficient approach to solving the road network equilibrium traffic assignment problem. **Transportation Research**, Elsevier, v. 9, n. 5, p. 309–318, 1975.

LEVY, N.; BEN-ELIA, E. Emergence of system optimum: A fair and altruistic agent-based route-choice model. **Procedia Computer Science**, v. 83, p. 928–933, 2016. ISSN 18770509.

LEYTON-BROWN, K.; SHOHAM, Y. **Essentials of Game Theory: A Concise Multidisciplinary Introduction**. 1. ed. San Rafael, USA: Morgan and Claypool Publishers, 2008. 88 p. (Synthesis Lectures on Artificial Intelligence and Machine Learning, v. 2).

LI, M.; HUANG, H.-J. A regret theory-based route choice model. **Transportmetrica A: Transport Science**, v. 13, n. 3, p. 250–272, 2017.

LIN, H. et al. Braess's paradox, Fibonacci numbers, and exponential inapproximability. In: **Automata, Languages and Programming, 32nd International Colloquium, ICALP**. Springer, 2005. p. 497–512.

LITTLESTONE, N.; WARMUTH, M. The weighted majority algorithm. **Information and Computation**, v. 108, n. 2, p. 212–261, 1994.

LITTMAN, M. L. Markov games as a framework for multi-agent reinforcement learning. In: **Proceedings of the 11th International Conference on Machine Learning**. New Brunswick, NJ: Morgan Kaufmann, 1994. p. 157–163.

LITTMAN, M. L. Friend-or-Foe Q-learning in general-sum games. In: **Proceedings of the Eighteenth International Conference on Machine Learning (ICML01)**. San Francisco, CA, USA: Morgan Kaufmann, 2001. p. 322–328.

LOOMES, G.; SUGDEN, R. Regret theory: An alternative theory of rational choice under uncertainty. **The Economic Journal**, v. 92, n. 368, p. 805–824, 1982.

LUJAK, M.; GIORDANI, S.; OSSOWSKI, S. Route guidance: Bridging system and user optimization in traffic assignment. **Neurocomputing**, v. 151, p. 449–460, mar 2015. ISSN 09252312.

MCFADDEN, D. Disaggregate behavioral travel demand's RUM side: A 30-year retrospective. In: HENSHER, D. A. (Ed.). **Travel Behaviour Research: the Leading Edge**. Oxford: Elsevier, Pergamon, 2001. p. 17–63.

MEIR, R.; PARKES, D. C. When are marginal congestion tolls optimal? In: BAZZAN, A. L. C. et al. (Ed.). **Proceedings of the Ninth Workshop on Agents in Traffic and Transportation (ATT-2016)**. New York: CEUR-WS.org, 2016. p. 8. ISSN 1613-0073. Available from Internet: <http://ceur-ws.org/Vol-1678/paper3.pdf>. Accessed September 1, 2017.

MITCHELL, W. J.; BORRONI-BIRD, C. E.; BURNS, L. D. **Reinventing the Automobile**. Cambridge, MA: MIT Press, 2010.

NASH, J. **Non-Cooperative Games**. Thesis (PhD) — Priceton University, 1950.

NATIONAL SURFACE TRANSPORTATION INFRASTRUCTURE FINANCING COMMISSION. **Paying our way: A new framework for transportation finance**. Washington DC, 2009. Available from Internet: <https://itif.org/publications/2009/02/24/paying-our-way-new-framework-transportation-finance>. Accessed March 15, 2018.

NEW CITIES FOUNDATION. **Connected Commuting: Research and Analysis on the New Cities Foundation Task Force in San Jose**. [S.l.], 2012. Available from Internet: <http://www.newcitiesfoundation.org/wp-content/uploads/New-Cities-Foundation-Connected-Commuting-Full-Report.pdf>. Accessed December 6, 2016.

NISAN, N. et al. **Algorithmic Game Theory**. New York, NY, USA: Cambridge University Press, 2007. ISBN 0521872820.

NOWÉ, A.; VRANCX, P.; HAUWERE, Y.-M. D. Game theory and multi-agent reinforcement learning. In: ____. **Reinforcement Learning: State-of-the-Art**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 441–470. ISBN 978-3-642-27645-3.

ORTÚZAR, J. d. D.; WILLUMSEN, L. G. **Modelling transport**. 4. ed. Chichester, UK: John Wiley & Sons, 2011.

PALMA, A. d.; LINDSEY, R. Traffic congestion pricing methodologies and technologies. **Transportation Research Part C: Emerging Technologies**, v. 19, n. 6, p. 1377–1399, 2011.

PAPADIMITRIOU, C.; TSITSIKLIS, J. N. The complexity of markov decision processes. **Mathematics of Operations Research**, INFORMS, Linthicum, Maryland, USA, v. 12, n. 3, p. 441–450, August 1987.

PATHANIA, D.; KARLAPALEM, K. Social network driven traffic decongestion using near time forecasting. In: **Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems**. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2015. (AAMAS '15), p. 1761–1762. ISBN 978-1-4503-3413-6.

PEETA, S.; YU, J. W. A hybrid model for driver route choice incorporating en-route attributes and real-time information effects. **Networks and Spatial Economics**, v. 5, p. 21–40, 2005.

PIGOU, A. **The Economics of Welfare**. London: Palgrave Macmillan, 1920. (Palgrave Classics in Economics).

PRABUCHANDRAN, K. J.; BODAS, T.; TULABANDHULA, T. Reinforcement learning algorithms for regret minimization in structured markov decision processes (extended abstract). In: THANGARAJAH, J.; TUYLS, K. (Ed.). **Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016)**. Singapore: IFAAMAS, 2016. p. 1289–1290. ISBN 9781450342391.

PROPER, S.; TUMER, K. Multiagent learning with a noisy global reward signal. In: ORGANIZATION. **Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence**. Bellevue: AAAI press, 2013. p. 826–832.

RAMOS, G. de. O.; BAZZAN, A. L. C. Towards the user equilibrium in traffic assignment using GRASP with path relinking. In: **Proceedings of the 2015 on Genetic and Evolutionary Computation Conference (GECCO '15)**. New York, NY, USA: ACM, 2015. p. 473–480. ISBN 978-1-4503-3472-3. Available from Internet: <http://doi.org/10.1145/2739480.2754755>. Accessed July 12, 2015.

RAMOS, G. de. O.; BAZZAN, A. L. C. Efficient local search in traffic assignment. In: **2016 IEEE Congress on Evolutionary Computation (CEC)**. Vancouver: IEEE, 2016. p. 1493–1500. ISBN 9781509006229. Available from Internet: <http://doi.org/10.1109/CEC.2016.7743966>. Accessed March 15, 2018. Accessed June 30, 2016.

RAMOS, G. de. O.; BAZZAN, A. L. C.; SILVA, B. C. da. Analysing the impact of travel information for minimising the regret of route choice. **Transportation Research Part C: Emerging Technologies**, v. 88, p. 257–271, Mar 2018. ISSN 0968-090X. Available from Internet: <http://doi.org/10.1016/j.trc.2017.11.011>. Accessed March 15, 2018.

RAMOS, G. de. O.; GRUNITZKI, R. An improved learning automata approach for the route choice problem. In: KOCH, F.; MENEGUZZI, F.; LAKKARAJU, K. (Ed.). **Agent Technology for Intelligent Mobile Services and Smart Societies**. Springer Berlin Heidelberg, 2015, (Communications in Computer and Information Science, v. 498). p. 56–67. ISBN 978-3-662-46240-9. Available from Internet: <http://doi.org/10.1007/978-3-662-46241-6_6>. Accessed January 5, 2015.

RAMOS, G. de. O.; SILVA, B. C. da; BAZZAN, A. L. C. Learning to minimise regret in route choice. In: DAS, S. et al. (Ed.). **Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017)**. São Paulo: IFAAMAS, 2017. p. 846–855. Available from Internet: <http://ifaamas.org/Proceedings/aamas2017/pdfs/p846.pdf>. Accessed May 12, 2017.

RASHIDI, T. H. et al. Exploring the capacity of social media data for modelling travel behaviour: Opportunities and challenges. **Transportation Research Part C: Emerging Technologies**, v. 75, p. 197–211, 2017.

RIETVELD, P. The economics of information in transport. **Tinbergen Institute Discussion Paper 10-110/3**, p. 1–25, nov 2010. Available from Internet: <http://doi.org/10.2139/ssrn.1702903>. Accessed March 15, 2018.

ROBBINS, H. Some aspects of the sequential design of experiments. **Bulletin of the American Mathematical Society**, v. 58, n. 5, p. 527–535, 1952.

ROSENTHAL, R. W. A class of games possessing pure-strategy Nash equilibria. **International Journal of Game Theory**, v. 2, p. 65–67, 1973.

ROUGHGARDEN, T. **Selfish Routing and the Price of Anarchy**. Cambridge: MIT Press, 2005. ISBN 0262182432.

ROUGHGARDEN, T. On the severity of Braess's paradox: Designing networks for selfish users is hard. **Journal of Computer and System Sciences**, v. 72, n. 5, p. 922–953, 2006. ISSN 0022-0000.

ROUGHGARDEN, T. Routing games. In: NISAN, N. et al. (Ed.). **Algorithmic game theory**. [S.l.]: Cambridge University Press, 2007. p. 461–486. ISBN 9780521872829.

ROUGHGARDEN, T.; TARDOS, É. How bad is selfish routing? **Journal of the ACM**, v. 49, n. 2, p. 236–259, 2002.

SANDHOLM, T. Perspectives on multiagent learning. **Artificial Intelligence**, v. 171, n. 7, p. 382–391, May 2007.

SHARON, G. et al. Real-time adaptive tolling scheme for optimized social welfare in traffic networks. In: DAS, S. et al. (Ed.). **Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017)**. São Paulo: IFAAMAS, 2017. p. 828–836.

SHEFFI, Y. **Urban Transportation Networks: Equilibrium Analysis With Mathematical Programming Methods**. Englewood Cliffs, USA: Prentice-Hall, 1984. ISBN 0139397299.

SHOHAM, Y.; POWERS, R.; GRENAGER, T. If multi-agent learning is the answer, what is the question? **Artificial Intelligence**, Elsevier Science, Essex, UK, v. 171, n. 7, p. 365–377, May 2007.

STEFANELLO, F.; BAZZAN, A. L. C. **Traffic Assignment Problem - Extending Braess Paradox**. Porto Alegre, RS, 2016. 24 p. Available from Internet: <www-usr.inf.ufsm.br/~stefanello/publications/Stefanello2016Braess.pdf>. Accessed March 15, 2018.

STEFANELLO, F.; SILVA, B. C. da; BAZZAN, A. L. C. Using topological statistics to bias and accelerate route choice: preliminary findings in synthetic and real-world road networks. In: **Proceedings of Ninth International Workshop on Agents in Traffic and Transportation**. New York, USA: [s.n.], 2016. p. 1–8. Available from Internet: <http://ceur-ws.org/Vol-1678/paper11.pdf>. Accessed June 30, 2016.

STONE, P.; VELOSO, M. Multiagent systems: A survey from a machine learning perspective. **Autonomous Robots**, v. 8, n. 3, p. 345–383, July 2000.

SUTTON, R.; BARTO, A. **Reinforcement Learning: An Introduction**. Cambridge, MA: MIT Press, 1998.

TAVARES, A. R.; BAZZAN, A. L. An agent-based approach for road pricing: system-level performance and implications for drivers. **Journal of the Brazilian Computer Society**, Springer, v. 20, n. 1, p. 15, 2014.

TUYLS, K.; WEISS, G. Multiagent learning: Basics, challenges, and prospects. **AI Magazine**, v. 33, n. 3, p. 41–52, 2012.

VASSERMAN, S.; FELDMAN, M.; HASSIDIM, A. Implementing the wisdom of waze. In: **Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI**. [s.n.], 2015. p. 660–666.

VERBEECK, K. et al. Exploring selfish reinforcement learning in repeated games with stochastic rewards. **Autonomous Agents and Multi-Agent Systems**, v. 14, n. 3, p. 239–269, Apr 2007. ISSN 1387-2532.

VRANCX, P.; VERBEECK, K.; NOWÉ, A. Learning to take turns. In: **Proceedings of the AAMAS 2010 Workshop on Adaptive Learning Agents and Multi-Agent Systems (ALA 2010)**. [S.l.: s.n.], 2010. p. 1–7.

WANG, G.; MA, S.; JIA, N. A combined framework for modeling the evolution of traveler route choice under risk. **Transportation Research Part C: Emerging Technologies**, v. 35, p. 156–179, 2013.

WARDROP, J. G. Some theoretical aspects of road traffic research. **Proceedings of the Institution of Civil Engineers, Part II**, v. 1, n. 36, p. 325–362, 1952.

WATKINS, C. J. C. H.; DAYAN, P. Q-learning. **Machine Learning**, Kluwer Academic Publishers, Hingham, MA, USA, v. 8, n. 3, p. 279–292, 1992. ISSN 0885-6125.

WAUGH, K. et al. Solving games with functional regret estimation. In: **Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence**. [S.l.]: AAAI Press, 2015. p. 2138–2144.

WOLPERT, D. H.; TUMER, K. **An Introduction to Collective Intelligence**. [S.l.], 1999. 88 p. ArXiv:cs/9908014 [cs.LG].

WOLPERT, D. H.; TUMER, K. Collective intelligence, data routing and braess' paradox. **Journal of Artificial Intelligence Research**, v. 16, p. 359–387, 2002.

YANG, H.; MENG, Q.; LEE, D.-H. Trial-and-error implementation of marginal-cost pricing on networks in the absence of demand functions. **Transportation Research Part B: Methodological**, v. 38, n. 6, p. 477–493, jul 2004.

YE, H.; YANG, H.; TAN, Z. Learning marginal-cost pricing via a trial-and-error procedure with day-to-day flow dynamics. **Transportation Research Part B: Methodological**, v. 81, p. 794–807, nov 2015.

YEN, J. Y. Finding the k shortest loopless paths in a network. **Management Science**, v. 17, n. 11, p. 712–716, 1971. Available from Internet: <http://pubsonline.informs.org/doi/abs/10.1287/mnsc.17.11.712>. Accessed December 19, 2013.

ZHANG, J. et al. Data-driven intelligent transportation systems: A survey. **IEEE Transactions on Intelligent Transportation Systems**, v. 12, n. 4, p. 1624–1639, Dec 2011. ISSN 1524-9050.

ZHANG, L. et al. Online bandit learning for a special class of non-convex losses. In: BONET, B.; KOENIG, S. (Ed.). **Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence**. Austin: AAAI Press, 2015. p. 3158–3164.

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

ZINKEVICH, M. et al. Regret minimization in games with incomplete information. In: PLATT, J. C. et al. (Ed.). **Advances in Neural Information Processing Systems 20**. [S.l.]: Curran Associates, Inc., 2008. p. 1729–1736.

APPENDIX A — COMPLEXITY ANALYSIS OF THE ALGORITHMS

In this appendix, we present the procedure used to simulate our algorithms and analyse its complexity. As input, this procedure receives an instance of the P of the route choice problem, the set of the K shortest routes for each OD pair, and some parameters (i.e., the number of episodes to run T and the learning and exploration decay rates λ and μ). Firstly, the procedure initialises all agents' Q-tables and, in the case of Algorithms 3.1 and 4.1, their history of reward estimates. Afterwards, the procedure simulates one episode a time, up to episode T. Simulating an episode involves the three steps: (i) letting agents choose their routes, (ii) updating the travel time on the routes, and (iii) providing the travel time to the agents so that they can update their policies. The complete procedure is presented in Algorithm A.1.

The second step of the algorithm updates the travel time of all routes according to the current choices of the agents. The complexity analysis of this process and some of its implementation details are presented in the next proposition. To enhance presentation, hereafter we use $d = |D|$, $l = |L|$, and m to denote the number of drivers, links, and OD pairs, respectively.

Algorithm A.1: Simulation procedure

input: instance $P = (G, D, f)$ of the route choice problem,
set of K shortest routes A_i for all agents with $\mathcal{A} = \{A_i \mid i \in D\}$,
number of episodes T,
learning decay rate λ,
exploration decay rate μ

for $i \in D$ **do**
  initialise Q-table of agent i as: $Q(a_i) \leftarrow 0 \; \forall a_i \in A_i$;
end
for $t \in \{1, \ldots, T\}$ **do**
  `// step (i): agents choose their actions`
  for $i \in D$ **do**
    $\dot{a}_i^t \leftarrow$ agent i chooses its action;
  end
  `// step (ii): update routes' travel time`
  update travel time on all links (and routes);
  `// step (iii): agents observe travel time and update their policies`
  for $i \in D$ **do**
    inform agent i about its travel time $f_{\dot{a}_i^t}$;
  end
end

Proposition A.1. *Updating the travel time of all routes, i.e., step (ii) of Algorithm A.1, has $O(l(d + mK))$ time complexity, for d drivers, l links, m OD pairs, and K routes per OD pair.*

Proof. Given the problem instance $P = (G, D, f)$, we represent each link by means of its ID, which we assume that is provided with the graph G. Considering that the set of routes is also given as input, the links themselves are irrelevant here (i.e., only their IDs are necessary). We also assume that the links' current flow and travel time are represented by means of two arrays, whose space complexity is $O(|L|)$.

Updating the travel time of all routes involves the following steps: (a) updating the flow on all links, (b) updating the travel time on all links, and (c) updating the travel time on all routes.

Step (a) involves two parts. Initially, we need to set the flow on all links to zero, which involves $O(l)$ updates. Then, we need to set the current flow on all links. To this end, we need to iterate over all agents and, for each agent, iterate over all links of its current route, adding 1 unit to the flow of that link. A route has at most $O(l)$ links, so this second part takes $O(dl)$. The complete step then has a complexity of $O(dl)$.

Step (b) iterates over all links, updating the travel time on that link using the VDF f provided with the problem instance P. This process has a time complexity of $O(l)$.

Step (c) iterates over all routes and, for each route, sums the travel time of the links comprising it. Every route has at most $O(l)$ links and we have K routes for each of the m OD pairs. The time complexity here is $O(mKl)$.

Summing all steps, we obtain a time complexity of $O(dl + l + mKl)$ or simply $O(l(d + mK))$. □

The other steps of Algorithm A.1 are performed in different ways, depending on which approach we want to simulate, i.e., Algorithm 3.1 or Algorithm 4.1 or and Algorithm 5.1. To be precise, the differences are:

- When simulating our regret-minimising Q-learning algorithm without the app information (from Chapter 3), step (i) corresponds to lines 4–5 of Algorithm 3.1 and step (iii) corresponds to lines 6–12 of Algorithm 3.1.
- When simulating our regret-minimising Q-learning algorithm with the app information (from Chapter 4), step (i) corresponds to lines 4–6 of Algorithm 4.1 and step (iii) corresponds to lines 7–13 of Algorithm 4.1.
- When simulation our toll-based Q-learning (from Chapter 5), step (i) corresponds

to lines 3–4 of Algorithm 5.1 and step (iii) corresponds to lines 5–7 of Algorithm 5.1.

We can now analyse the complexity of our approaches. In what follows, we restate and prove Propositions 3.1, 4.1, and 5.2.

Proposition 3.1. *Our regret-minimising Q-learning approach has $O(T(dK+ld+lmK))$ time complexity and $O(dK)$ space complexity, for T episodes, d drivers, and K actions, l links, and m OD pairs.*

Proof. In order to analyse our algorithm, we firstly analyse the complexity of the agents' initialisation. Afterwards, we analyse the episode loop of the simulation procedure (lines 4–12 of Algorithm A.1) and then sum all up to obtain the overall complexity.

We implement an agent's Q-table using an array. In this sense, all Q-tables are initialised in $O(dK)$ time complexity and, in total, they have $O(dK)$ space complexity. An agent's history of estimates, on the other hand, can be implemented more efficiently by means of two arrays: one for the accumulated estimated reward of each action and another for the most recent reward observation of each action. In this way, we eliminate the need for storing all reward estimates for every action along time. Considering that d agents are being simulated, these histories demand a $O(dK)$ time complexity for initialisation and $O(dK)$ space complexity. Observe that, using these data structures, an agent's Q-table and history of estimates remain fixed throughout the simulation.

Step (i) of Algorithm A.1 corresponds to lines 4–5 of Algorithm 3.1. We firstly consider the complexity associated with each agent. The learning and exploration rates are updated in constant time. The ϵ-greedy exploration scheme involves generating a random number in $O(1)$ time and then selecting the best action (if the random number is larger than ϵ) or a random action (if the random number is lower than ϵ). The ϵ-greedy scheme receives a Q-table as input, which we implemented as an array (as described in the beginning of this proof). In this sense, an action can be drawn uniformly at random in constant time and the best one can be selected in $O(K)$, and thus ϵ-greedy has $O(K)$ time complexity and constant space complexity. Hence, for each agent, step (i) has a $O(K)$ time complexity and $O(1)$ space complexity, which, considering that there are d agents, becomes $O(dK)$ time complexity and $O(d)$ space complexity (just to store their choices).

Step (ii) has a time complexity of $O(l(d + mK))$, as given by Proposition A.1.

Step (iii) of Algorithm A.1 corresponds to lines 6–12 of Algorithm 3.1. The reward estimate of an action can be updated in constant time, which accounts for a $O(K)$ time

complexity for all actions of the agent. Using the data structures defined in the beginning of this proof, the action regret of the selected action of each agent can be computed in $O(K)$ time for the first term and $O(1)$ time for the second term, thus summing to $O(K)$ for each agent. The other lines of step (iii) can be computer in constant time. Due to the data structures used, no additional space is required here. Hence, considering the existence of d agents, step (iii) translates into $O(dK)$ time complexity and $O(1)$ space complexity.

A complete episode has then a time complexity of $O(dK + l(d + mK) + dK) \in O(dK + ld + lmK)$ and a space complexity of $O(d)$. Putting all together, and considering T episodes, the complete algorithm has a time complexity of $O(dK + T(dK + ld + lmK)) \in O(T(dK + ld + lmK))$. As for the space complexity, considering that the data structures used for the Q-table and history of estimates remain constant along time, the space complexity is $O(dK + d) \in O(dK)$. □

Proposition 4.1. *Our regret-minimising Q-learning with app information approach has $O(T(dK + ld + lmK))$ time complexity and $O(dK)$ space complexity, for T episodes, d drivers, and K actions, l links, and m OD pairs.*

Proof. The analysis here is basically the same as in the proof of Proposition 3.1 above. The only difference is that we have the app informations. The app information can be stored in an array with space complexity of $O(mK)$. Observe that this single array is enough to store the recommendations for all drivers, regardless of their OD pairs. This array can be populated during the step (ii) of Algorithm A.1 without affecting Proposition A.1.

On the agents' side, the app information is used only for updating the action regret of the currently chosen action. In this sense, it does not affect the time complexity of the agents. The space complexity also remains unaffected, since the app information are not stored by the agent.

So far, by using the app, the only difference to Proposition 3.1 is the presence of an additional mK term in the space complexity, i.e., $O(mK + dK)$. Nevertheless, observe that the number of OD pairs m cannot be higher than the number of drivers. By definition, a driver has exactly one OD pair and an OD pair has multiple drivers. In this sense, $m \leq d$ and, thus, $O(mK + dK) \in O(dK)$.

Therefore, when the app is used, the time and space complexity of our approach remain $O(T(dK + ld + lmK))$ and $O(dK)$, respectively. □

Proposition 5.2. *Our toll-based Q-learning approach has $O(T(dK + ld + lmK))$ time complexity and $O(dK)$ space complexity, for T episodes, d drivers, and K actions, l links and m OD pairs.*

Proof. The complexity analysis of the toll-based Q-learning is similar to that of previous algorithms. Again, we implement an agent's Q-table using an array so that all Q-tables can be initialised in $O(dK)$ time complexity. The Q-tables have a fixed size and, in total, they have a $O(dK)$ space complexity.

As in the previous proofs, we first analyse the complexity of the episode loop of the simulation procedure (lines 4–12 of Algorithm A.1) and then sum all up to obtain the overall complexity.

Step (i) of Algorithm A.1 corresponds to lines 3–4 of Algorithm 5.1. This step is exactly the same as in Algorithm 3.1. Hence, it has $O(dK)$ time complexity and $O(d)$ space complexity (which is required for storing the agents' choices).

Step (ii) has a time complexity of $O(l(d + mK))$, as given by Proposition A.1.

Step (iii) of Algorithm A.1 corresponds to lines 5–7 of Algorithm 5.1. These lines are executed in constant time, which, considering that this step is performed by d agents, translates into $O(d)$ time complexity. As for space, considering that the Q-table has a constant size, the space complexity of this step remains constant.

A complete episode has then a time complexity of $O(dK + l(d + mK) + d) \in O(dK + ld + lmK)$ and a space complexity of $O(d)$. Putting all together, and considering T episodes, the complete algorithm has a time complexity of $O(dK + T(dK + ld + lmK)) \in O(T(dK + ld + lmK))$. As for space, considering that the Q-table has a fixed size, we have $O(dk + d) \in O(dk)$ space complexity. □

APPENDIX B — FIGURES OF THE ROAD NETWORKS

In this appendix, we present the topology of the road networks used in our experiments. The complete description of these networks, including VDFs, is available at <https://github.com/maslab-ufrgs/network-files>. Considering that some routes are very similar (e.g., Braess graphs), here we only present the most representative ones. Specifically, we present the following networks: B^3 (Figure B.1), BB^3 (Figure B.2), OW (Figure B.3), and SF (Figure B.4).

Figure B.1: B^3 network

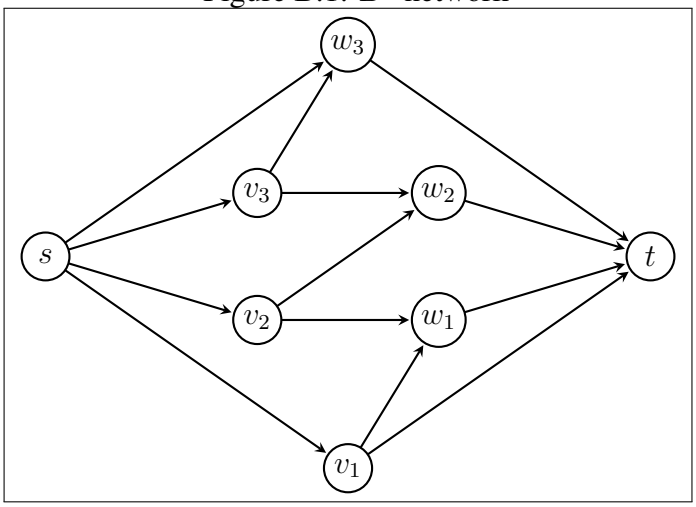


Figure B.2: BB^3 network

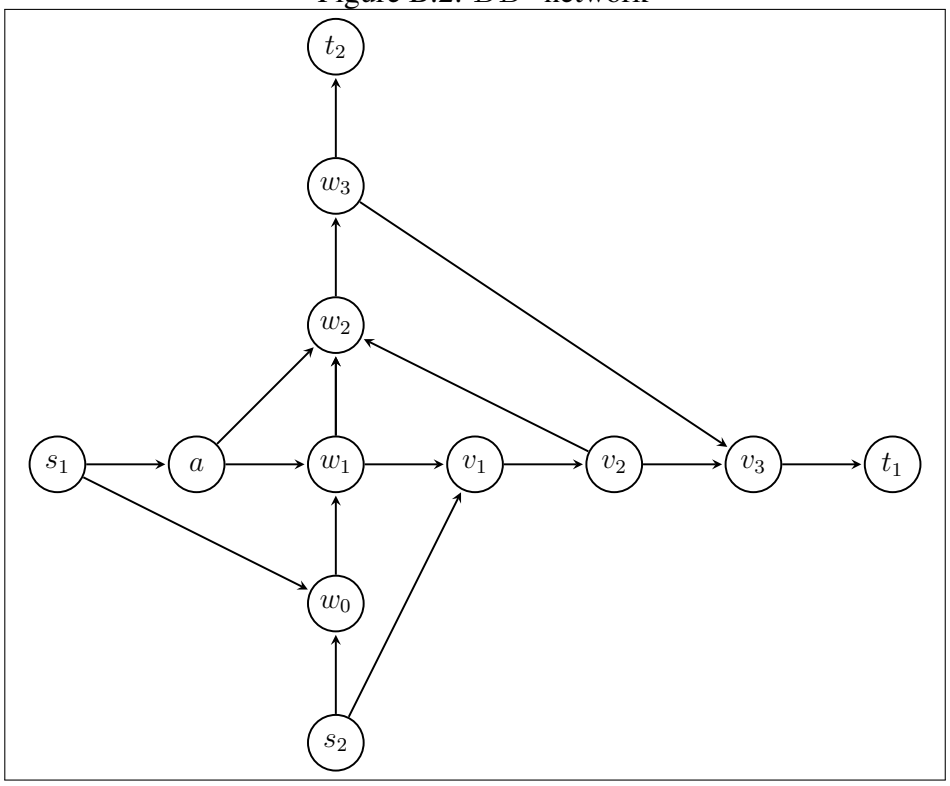


Figure B.3: OW network

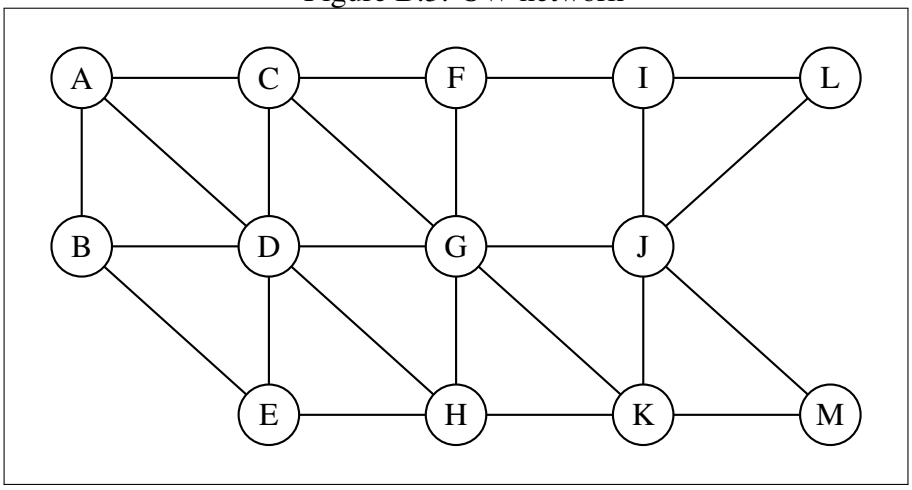
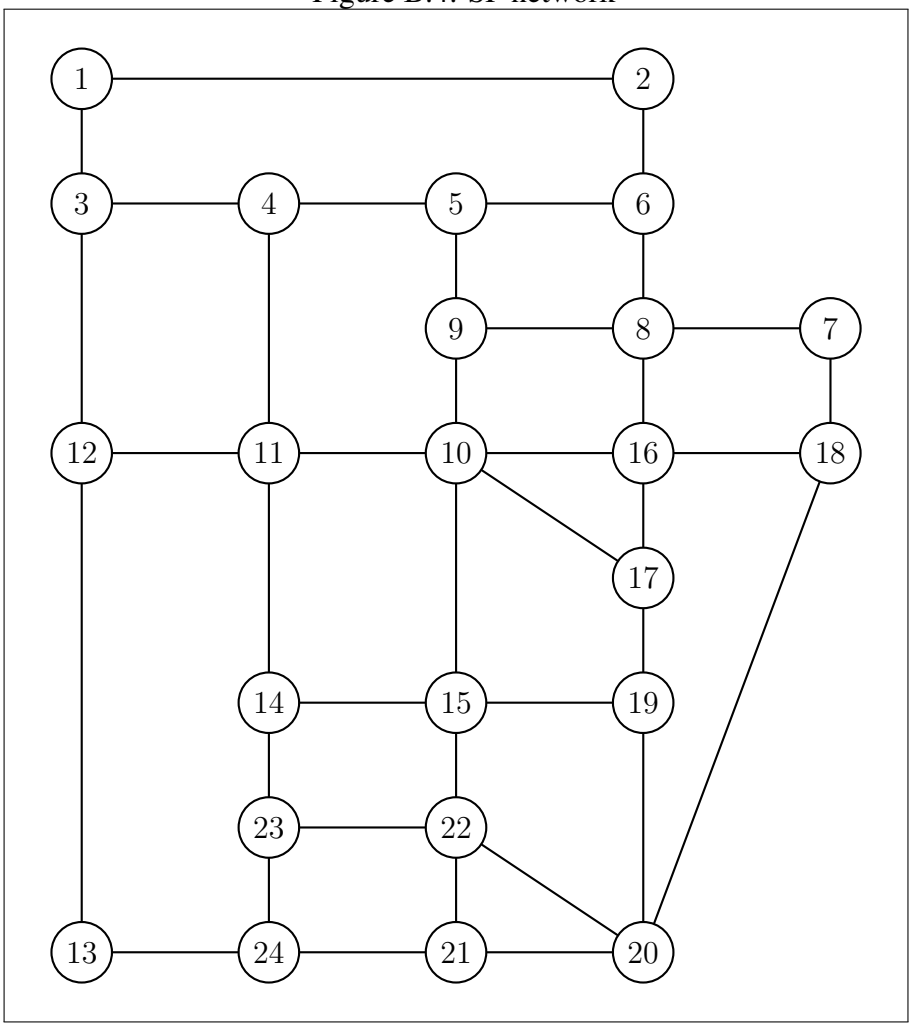


Figure B.4: SF network



APPENDIX C — RESUMO ESTENDIDO EM PORTUGUÊS

C.1 Motivação

Mobilidade urbana eficiente desempenha um papel fundamental na sociedade moderna. No entanto, a crescente demanda por tal mobilidade associada à ausência de investimentos adequados na oferta (infraestrutura viária) têm comprometido a eficiência dos sistemas viários existentes, como evidenciado pelos crescentes (em número e intensidade) congestionamentos. Segundo o *Centre for Economics and Business Research* (2014), o impacto de congestionamentos na economia dos Estados Unidos superou a marca de US$ 120 bilhões em 2013. Além do mais, tais custos devem crescer cerca de 50% até 2030.

Tradicionalmente, uma das práticas mais comuns para minimizar congestionamentos consiste em expandir a infraestrutura viária existente. Entretanto, tal abordagem tem se mostrado insustentável sob diversas perspectivas. Além do mais, como descrito pelo paradoxo de Braess (1968), a expansão da capacidade da infraestrutura viária pode até mesmo piorar o desempenho do tráfego. Desta forma, abordagens que fazem um uso mais eficiente da infraestrutura já existente têm se mostrado cada vez mais importantes.

Esta tese aborda o tráfego sob a perspectiva de motoristas que devem escolher as rotas que minimizam seus custos de viagem. Este é o chamado problema de escolha de rotas. Vale ressaltar que, neste contexto, os motoristas possuem um comportamento tipicamente egoísta, buscando escolher as rotas que melhoram seu desempenho independente do prejuízo que tais escolhas possam gerar aos demais. Neste trabalho, investigamos como tais motoristas podem aprender a escolher suas rotas de uma maneira eficiente, com base na experiência adquirida ao longo do tempo. Desta forma, o problema é abordado do ponto de vista de aprendizagem por reforço multiagente.

Aprendizagem por reforço multiagente (do inglês, MARL) é uma tarefa desafiadora em que agentes buscam, concorrentemente, aprender um comportamento (política) capaz de maximizar sua utilidade. Aprender neste tipo de cenário é difícil porque os agentes devem se adaptar uns aos outros, tornando o objetivo um alvo em movimento (BUŞONIU; BABUSKA; SCHUTTER, 2008; TUYLS; WEISS, 2012). Consequentemente, não existem garantias de convergência para problemas de MARL em geral. Por outro lado, resultados promissores têm sido alcançados para problemas específicos. A presente tese segue justamente esta direção, buscando fornecer garantias de convergên-

cia para o caso específico do problema de escolha de rotas. Em particular, esta tese busca garantir a convergência de algoritmos de MARL para o equilíbrio dos usuários (onde nenhum motorista consegue melhorar seu desempenho mudando de rota) e para o ótimo do sistema (onde o tempo médio de viagem é mínimo) (WARDROP, 1952).

C.2 Desafios

O principal objetivo desta tese é mostrar que, no contexto de escolha de rotas, é possível garantir a convergência de algoritmos de MARL, sob certas condições, tanto para o equilíbrio dos usuários quanto para o ótimo do sistema. Em particular, busca-se investigar como os agentes (motoristas) podem aprender por conta própria, eliminando certas suposições tipicamente feitas na literatura. Neste sentido, é possível identificar três desafios, no contexto de escolha de rotas, conforme abordado a seguir.

- *Sob quais circunstâncias é possível garantir que agentes convergirão para o equilíbrio dos usuários utilizando aprendizagem por reforço?* Conforme discutido anteriormente, quando diversos agentes precisam aprender uma política ótima simultaneamente em um ambiente compartilhado, temos que seu objetivo se torna um alvo em movimento. Este é justamente o caso do problema de escolha de rotas. Uma alternativa para lidar com este tipo de problema é através de algoritmos de minimização de arrependimento, que têm obtido resultados promissores em cenários multiagente (CESA-BIANCHI; LUGOSI, 2006). O *arrependimento* de um agente mede seu desempenho médio em comparação com o da melhor ação (rota) fixa. Um motorista que minimiza seu arrependimento acaba escolhendo a rota que minimiza seu custo de viagem. Trabalhos anteriores têm obtido bons resultados no contexto de aprendizagem por reforço (BOWLING, 2005; ZINKEVICH et al., 2008; WAUGH et al., 2015), *congestion games* (BLUM; EVEN-DAR; LIGETT, 2010), e *multi-armed bandits* (AUER et al., 2002; AWERBUCH; KLEINBERG, 2004). No entanto, tais trabalhos geralmente assumem que os agentes possuem conhecimento completo sobre as funções de custo. O desafio aqui, portanto, consiste em fornecer meios para que os agentes possam estimar seu arrependimento baseando-se exclusivamente em informações locais (ou seja, seu próprio conhecimento) e encontrar meios de garantir a convergência de tal abordagem para o equilíbrio dos usuários.

- *Sob quais circunstâncias é possível melhorar o desempenho dos agentes através do fornecimento de informações não-locais?* Uma extensão natural para o desafio anterior consiste em entender o efeito de informações não-locais no processo de aprendizagem do agente. De fato, tais informações podem ser facilmente obtidas atualmente (através de dispositivos de navegação, por exemplo) e poderiam ser utilizadas para estimar o arrependimento com mais precisão. Existem diversas pesquisas nesta linha. No entanto, geralmente assume-se que os agentes possuem conhecimento completo sobre as funções de custo (BEN-ELIA; ISHAQ; SHIFTAN, 2013) ou ainda que uma entidade central pode observar antecipadamente as decisões dos agentes (KLÜGL; BAZZAN, 2004; VASSERMAN; FELDMAN; HASSIDIM, 2015). Desta forma, o maior desafio neste sentido consiste em definir adequadamente a natureza das informações não-locais, bem como como combinar efetivamente informações locais e não-locais.
- *Sob quais circunstâncias é possível garantir que agentes convergirão para o ótimo do sistema utilizando aprendizagem por reforço?* Quando os agentes buscam minimizar seus custos de viagem, o sistema tende a convergir para um equilíbrio dos usuários. Porém, tal equilíbrio é ineficiente do ponto de vista global. Um comportamento altruísta por parte dos agentes poderia resolver este problema, porém não é possível forçar os agentes a se comportar desta forma (FEHR; FISCHBACHER, 2003). O uso de pedágios, por outro lado, pode ser utilizado justamente para esta finalidade (BECKMANN; MCGUIRE; WINSTEN, 1956). Um dos mecanismos mais conhecidos para tal é o pedágio de custo marginal, onde um agente é cobrado proporcionalmente ao custo que ele impõe nos demais (PIGOU, 1920). De fato, este mecanismo tem sido amplamente utilizado na literatura. Todavia, costuma-se assumir que os pedágios são computados por uma entidade central com conhecimento completo sobre o estado atual da rede viária (COLE; DODIS; ROUGHGARDEN, 2003; CHEN; KEMPE, 2008; SHARON et al., 2017). Portanto, o desafio aqui corresponde a definir tais mecanismos de uma forma decentralizada, eliminando as suposições de conhecimento completo.

C.3 Principais contribuições

Nesta tese, o problema de aprendizagem por reforço multiagente é analisado sob uma perspectiva teórica, no contexto de escolha de rotas. De modo a superar os desa-

fios elencados na seção anterior, são formuladas as seguintes hipóteses: (i) utilizar o arrependimento como sinal de reforço leva os agentes a convergir para o equilíbrio dos usuários, (ii) fornecer informações não-locais para os agentes melhora seu aprendizado, e (iii) cobrar pedágios de custo marginal leva os agentes a convergir para o ótimo do sistema. Com base nestas hipóteses, as contribuições desta tese são definidas em três frentes, descritas nas subseções a seguir.

C.3.1 Aprendizagem com base no arrependimento

Esta frente introduz uma variação do algoritmo *Q-learning* capaz de minimizar o arrependimento dos agentes. Em linhas gerais, o algoritmo estima o arrependimento associado com cada ação do agente e utiliza tal informação como sinal de reforço para atualizar os valores Q correspondentes. Diferentemente de outras abordagens disponíveis na literatura, nosso método depende apenas das recompensas observadas pelo agente. Com base neste algoritmo, foi realizada uma análise teórica do desempenho dos agentes, sendo possível estabelecer um limite superior no arrependimento e, consequentemente, garantir a convergência para um equilíbrio dos usuários aproximado, conforme detalhado nos teoremas a seguir.

Teorema 1 (adaptado do *Theorem 3.6*)**.** *O arrependimento do algoritmo até o tempo T é limitado superiormente por $O\left(\left(\frac{K-1}{TK}\right)\left(\frac{\mu^{T+1}-\mu}{\mu-1}\right)\right)$, onde K corresponde ao número de ações possíveis e μ denota o decaimento da taxa de exploração.*

Teorema 2 (adaptado do *Theorem 3.7*)**.** *O algoritmo converge para um equilíbrio dos usuários aproximado em ϕ, onde ϕ é o limite superior do arrependimento do algoritmo.*

De modo a validar os resultados teóricos, foram realizados ainda diversos de experimentos. Em particular, o algoritmo foi testado em diferentes instâncias do problema de escolha de rotas disponíveis na literatura e comparado com o algoritmo *Q-learning* padrão (que utiliza recompensas, ao invés do arrependimento, como sinal de reforço). Em suma, observou-se que nosso método foi capaz de reduzir o arrependimento em 21,5% (em média) quando comparado ao *Q-learning* padrão. Além do mais, constatou-se que o arrependimento obtido foi compatível com a análise teórica. Em relação à proximidade do equilíbrio, nosso método alcançou 99,38% do mesmo, enquanto o *Q-learning* padrão obteve 98,8%, ou seja, nosso método reduziu pela metade a distância para o equilíbrio.

C.3.2 Uso de informações não-locais

Dando continuidade à frente anterior, o algoritmo foi estendido para lidar com informações não-locais, ou seja, informações de viagem que podem ser fornecidas por dispositivos de navegação. Desta forma, foi definida uma entidade capaz de fornecer tais informações, chamada simplesmente de *app*. A noção de arrependimento foi então ajustada para considerar não apenas a informação local do agente, mas também a informação fornecida pelo *app*. Novamente, foram realizados diversos experimentos, mas desta vez para avaliar o desempenho dos agentes na presença do *app*. A partir destes experimentos, observou-se uma redução de 13,7% no arrependimento médio, significando que, ao usar as informações fornecidas pelo *app*, os agentes são capazes de estimar com maior precisão o arrependimento de suas ações, o que melhora seu desempenho. Em termos de proximidade do equilíbrio, o uso do *app* não trouxe melhoras significativas, dado que os resultados sem o *app* já estavam consideravelmente próximos do mesmo.

C.3.3 Aprendizagem com base em pedágios de custo marginal

Nas frentes anteriores, o objetivo foi analisar a convergência para um equilíbrio dos usuários. Porém, do ponto de vista global, tal resultado pode ser consideravelmente pior que o ótimo do sistema. Esta deterioração do desempenho global é uma consequência do comportamento egoísta dos agentes, sendo assim denominado preço da anarquia (PAPADIMITRIOU; TSITSIKLIS, 1987). De modo a reduzir o impacto do comportamento egoísta dos agentes, a terceira frente desta tese introduz um mecanismo de pedágios de custo marginal, onde os agentes são cobrados proporcionalmente ao custo imposto por eles aos demais (PIGOU, 1920). O mecanismo de pedágios proposto é genérico (compreendendo as principais funções de custo utilizadas na literatura) e é cobrado apenas ao término de cada viagem (podendo ser computado pelos próprios agentes e utilizado como parte do sinal de reforço para atualizar os valores Q). Através de uma análise teórica do algoritmo, foi possível provar que o mesmo converge para o ótimo do sistema e que o valor dos pedágios é mais justo que o definido por outros métodos disponíveis na literatura,

Teorema 3 (adaptado do *Theorem 5.2* e do *Corollary 5.2*)**.** *O algoritmo converge para o ótimo do sistema no limite, minimizando assim o preço da anarquia.*

Teorema 4 (adaptado do *Theorem 5.4*)**.** *O mecanismo de pedágio a posteriori proposto é mais justo que mecanismos a priori existentes na literatura.*

Bem como nas frentes anteriores, diversos experimentos foram conduzidos para testar o método proposto. Em suma, constatou-se que o mesmo alcançou 99,95% do ótimo do sistema, em média, corroborando com a análise teórica de convergência.

C.4 Conclusões

Esta tese investigou o problema de escolha de rotas sob a perspectiva de aprendizagem por reforço multiagente. Este tipo de abordagem é bastante complexa, dado que os agentes precisam aprender uma política ótima concorrentemente em um ambiente com recursos limitados e compartilhados. Considerando este aspecto não-estacionário, não existem garantias de convergência para o problema geral de aprendizagem por reforço multiagente. Desta forma, esta tese buscou estabelecer tais garantias de convergência para o caso específico do problema de escolha de rotas.

Foram consideradas duas faces do problema: o equilíbrio dos usuários (que resulta do comportamento egoísta dos motoristas) e o ótimo do sistema (que corresponde ao melhor resultado possível do ponto de vista global). Para o primeiro caso, foi introduzido um algoritmo de minimização do arrependimento que estima o arrependimento associado com cada ação do agente e utiliza tal informação para atualizar os valores Q correspondentes. Para o segundo caso, foi introduzido um mecanismo de pedágios por custo marginal, que pode ser computado pelos próprios agentes e utilizado como parte do seu sinal de reforço. Em ambos os casos, foram realizadas extensas análises teóricas, permitindo estabelecer garantias de convergência. Além do mais, os resultados teóricos foram validados experimentalmente em diversas instâncias do problema de escolha de rotas disponíveis na literatura.

Como trabalhos futuros, diversas possibilidades se mostram promissoras. Primeiramente, formulações de arrependimento mais sofisticadas podem ser definidas para utilizar mais eficientemente o conhecimento dos agentes, acelerando assim o processo de aprendizagem. Em termos de análises, o próximo passo consiste em analisar a taxa de convergência dos algoritmos, permitindo entender melhor o processo de aprendizagem em diferentes instâncias do problema. No que se refere ao mecanismo de pedágios, seria interessante projetar mecanismos que permitam aos agentes antecipar o pedágio devido antes de suas viagens. Finalmente, almeja-se ainda estender as análises atuais para problemas mais genéricos considerando, por exemplo, situações em que os agentes não conhecem suas rotas antecipadamente.